Deep In-Situ Learning for Object Recognition

By

MIGUEL LAGUNES-FORTIZ

Department of Aerospace Engineering
UNIVERSITY OF BRISTOL

A dissertation submitted to the University of Bristol in accordance with the requirements of the degree of DOCTOR OF PHILOSOPHY in the Faculty of Engineering.

OCTOBER 2019

Word count: 40,338

ABSTRACT

This dissertation is about *in-situ* object recognition, meaning that *specific* objects (*instances*) can be learned from a few training examples that depict them within the place where such objects are commonly present or being used. Learning to recognize objects in-situ opposes to conventional approaches in deep learning of relying on large-scale class-level datasets of grouped instances, utilizing complex image acquisition setups or utilizing synthetic data.

We aim for a scalable, robust, and real-time system based on Convolutional Neural Networks (CNNs) that learn discriminative features from images depicting objects from an egocentric point of view. We are particularly interested in learning objects from a few examples taken directly by an agent or by a demonstrator, and where the CNN does not need a finetuning process for learning additional instances, motivated by the computational limitations in most autonomous platforms. We hope our approach will be helpful for robotic tasks such as object manipulation, human-robot interaction, semantic mapping, scene understanding, autonomous navigation, and contribute to FARSCOPE's vision on advancing the *state-of-the-art* of autonomy in robots and intelligent systems.

DEDICATION AND ACKNOWLEDGEMENTS

I want to thank my supervisors Prof. Walterio Mayol-Cuevas and Dr Dima Damen, for their patience and advice that allowed me to nourish a critical and curious mind. I am very grateful for teaching me not to lose sight of the big picture and for encouraging me to develop my research skills with the same emphasis as my technical expertise. Thank you to my examiners Prof. Andrew Calway and Prof. Toby Breckon for their fruitful advice and support.

I would like to thank the FARSCOPE staff and cohorts for all the welcoming support. Thanks to Salua, Krishna, Chanelle, Karam, Luis, Jay, Perla, Eduardo, Benny, Ramon, Abel and Obed, for the friendship and advice, making my doctoral studies an enjoyable and life-time experience. Thank you to all of the bright researchers in the VI-Lab and the Bristol Robotics Laboratory.

I dedicate this dissertation to my family, who has been my support and motivation all these years. To my parents Miguel and Sara, whom their love and hardworking life, has always inspired me to give the best of my efforts. To my siblings Ricardo and Erika and niece Gala, for their continuous support, cheering up and love despite the oceans between us. To my partner, Mikaela Webster, for all the love, advise and adventures that have made my days much brighter.

Finally, I want to thank CONACyT, the National Council of Science and Technology of Mexico, for its financial support during my doctoral studies with the scholarship number 686450. I will make my best effort to contribute with Mexico's scientific and technological development.

AUTHOR'S DECLARATION

I declare that the work in this dissertation was carried out in accordance with the requirements of the University's Regulations and Code of Practice for Research Degree Programmes and that it has not been submitted for any other academic award. Except where indicated by specific reference in the text, the work is the candidate's own work. Work done in collaboration with, or with the assistance of, others, is indicated as such. Any views expressed in the dissertation are those of the author.

SIGNED: ..... DATE: .....

TABLE OF CONTENTS

List of Tables

Table **Page**

List of Figures

List of Symbols

x	Vector with unnormalized (or not-yet normalized) prediction scores, produced by a classification layer. It is named *logits* vector within some deep learning frameworks such as TensorFlow.
$\hat{x}$	Embedding obtained from an intermediate layer of a Neural Network.
X	Input image fed to the Network.
$q(x)$	Softmax function applied to the vector x to computed a vector of probabilities.
$d(x_a, x_b)$	Metric distance between vectors x_a and x_b. In Chapters 5 and 6, corresponds to the Euclidean distance.
y	Vector containing grountruth labels.
$\hat{y}$	Vector containing the predicted labels from a classifier.
W	Matrix of trainable parameters in a Convolutional Neural Network.
h	Hidden state in a Recurrent Neural Network.
μ_a	Centroid in the features space of the object a.

CHAPTER 1

INTRODUCTION

"Everything must be made as simple as possible. But not simpler"

Albert Einstein

This thesis develops computer vision algorithms that learn to recognize *specific* objects (*instances*) from a small number of training examples collected and deployed in-situ. That is, from observations from where the objects currently are, perhaps after first encountering them, the algorithms immediately are able to recognize them again. We refer to this methodology as ***in-situ learning***, and it is an alternative to the conventional methodology of using complex mechanisms, such as rotating tables or synthetic images, that build a large-scale *class-level* dataset for training Convolutional Neural Networks (CNNs). We are particularly interested in learning objects from few examples taken directly by an *agent* or by a demonstrator, and where the CNN does not need a finetuning process for learning additional objects, motivated by the computational and energy limitations in most autonomous platforms. We depict this challenging problem in Figure 1.1.

The motivation of this thesis is two-fold: (1) As robots are starting to be used for solving problems in unstructured and dynamic environments, there is a need for them in having a high level of autonomy, this is, without requiring human intervention when they are performing their duties and when they face a novel problem to solve at hand. (2)

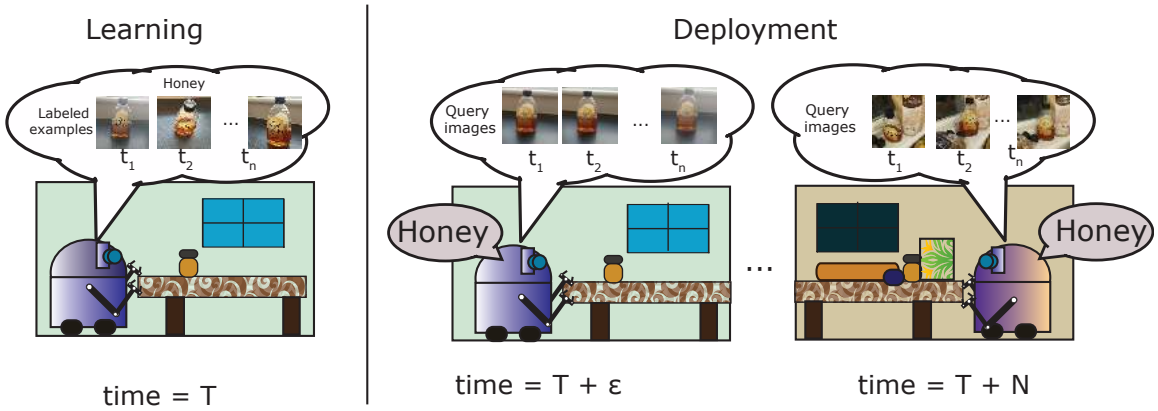
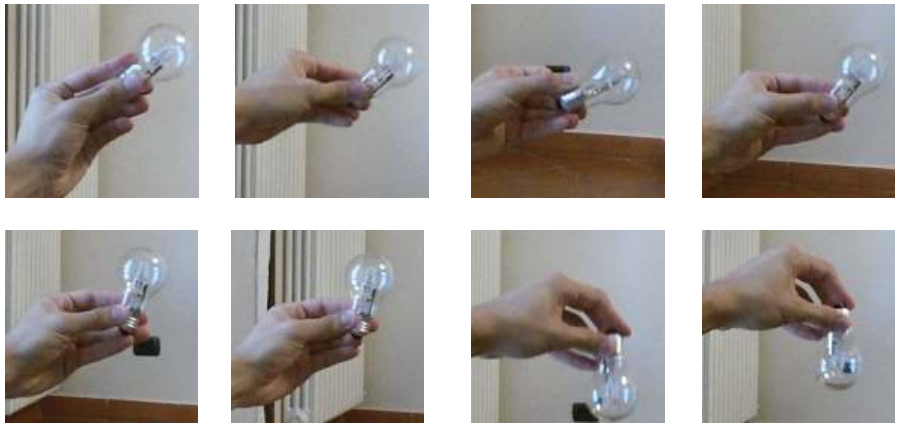


Figure 1.1: An agent learning to recognize a *specific* object from a few data collected *in-situ*, with scalable, robust, real-time performance, and onboard learning capabilities. Learning is performed at time T and deployed within seconds ($T+\epsilon$). The model should generalize to an unseen condition such as changing illumination, clutter, and occlusions subsequently encountered at ($T+N$). Image best viewed in the digital version.

Despite the progress in object recognition by the introduction of CNNs, designing models that can learn generalizable representations from a few examples is still a challenge far from being solved, and that has less attention in the computer vision community than approaches relying on large-scale datasets.

Related to the first point, our vision to increase the level of autonomy in robots is by giving them the capability of learning to recognize objects without the need of recurring to complex image-acquisition systems or gathering thousands of examples from the web. Instead, they utilize a few training examples, *i.e.,* low hundreds of training examples per instance, as contained in a few seconds of video data that an agent might collect when learning a new instance. We illustrate the contrast in the images collected between these approaches in Figure 1.2. On the in-situ scenario, an agent can record a short video clip by manipulating the object (like in the CORe50 dataset, Figure 1.2(b)). Noticeably, the background shows undesirable patterns such as curtains, wallpapers, bushes, etc. On the other hand, for the T-LESS dataset (Figure 1.2(a)), the illumination and background conditions are controlled, and a systematic acquisition process was performed, as opposed to the patternless image in CORe50. Noticeably, we aim to learning *specific* objects from a few training examples, and not learning *class-level* concepts using a few training examples from a single instance. For example, we do not aim to have a model that learns the concept of a *car*, by watching a single example of it. Additionally, our problem is less restrictive than the few-shot learning problem, where only a k number of samples

(a) T-LESS



(b) CORe50

Figure 1.2: Examples of data acquisition methods in public object recognition datasets, a controlled-environment as in T-LESS (a) and a more *in-situ*-like conditions as in Core50 (b).

(typically zero, one, or five) are used to train a model. We consider this more challenging scenario as a future direction of our work.

Related to the second motivational aspect, it is the interest in this thesis to get an understanding of the current limitations of CNNs for allowing them to learn to recognize objects in the way we, humans, do: being able to generalize from few data, with very little supervision and with the capability of learning new concepts efficiently, as well as forgetting irrelevant events, among other attributes (Margolis, 2015; Zenke et al., 2017; Lake et al., 2017).

1.1 Problem Description

We now make a description of the challenges we aim to solve, which involves the learning of generalizable visual representations from a few data, as shown in Figure 1.1. We identify three core challenges in the *in-situ learning* approach:

1. *Robustness*: During deployment, the learner has to deal with commonly encountered perturbations such as changes in perspective, illumination, ambiguous viewpoints, noise, and other nuisances.

2. *Domain Generalization*: The learner has to produce generalizable features from a few data that depicts objects within a specific environment.

3. *On-the-fly learning*: The learner has to accommodate new instances efficiently without forgetting the previously learned ones.

1.2 Approach

We aim for a generic instance-level object recognition approach, which does not rely on any assumptions about object properties such as texture, color, or geometry. The algorithms will use images from video data, depicting objects from an egocentric perspective at both training and deployment. For training, the images depict a single object that can be used for extracting, learning, and localizing relevant features. As for the stream during inference, it might depict clutter, occlusion, unseen viewpoints, and other circumstances not showing during training. Since our algorithms are meant to be used on robots and other mobile platforms, it is desired real-time rates during deployment (*e.g.,* in the magnitude of dozens of frames per second), and without requiring an enormous amount of computational resources such as multiple Graphical Processor Units (GPUs).

Our strategy consists of designing a discriminative model that can associate images depicting the same objects, even when they are from different environments. This is achieved by producing features which are close to each other for the same instance and separated otherwise. To deal with nuisances and ambiguous viewpoints, we propose the use of temporal filters that enforce the temporal coherence that exists in video data, which prevents sudden changes in the predictions between neighboring video frames.

1.3 Contributions

The contributions of this thesis are as follows:

- In Chapter 3, we propose the use of temporal filters for increasing the robustness of CNNs against commonly encountered nuisances. The filters consist of a simple sum rule and a machine learning approach that uses bidirectional LSTMs.
- In Chapter 4, we propose a multi-task CNN that classifies and reconstructs images simultaneously, the incorporation of the auxiliary image reconstruction task aims to achieve domain generalization for object recognition.
- In Chapter 5, we propose a discriminative CNN that classifies images and enforces discriminability on the learned features. Our discriminative model allows the learning of additional objects *on-the-fly*, by replacing the classification layer by the nearest neighbors search in the embeddings space.
- In Chapter 6, we propose utilizing the nearest centroids algorithm to accelerate the inference time, and to reduce the storage requirements. Additionally, we propose a novel regularization term to enforce discriminability between features and centroids. Furthermore, we utilize the distance between embeddings from neighboring video frames, to dynamically decide when to apply the sum-rule filter from Chapter 3. Finally, we propose a dataset for a systematic benchmarking of *in-situ* learning approaches.

1.4 Publications

A subset of the research work presented within this thesis has previously been published in the following peer-reviewed publications:

- *Lagunes Fortiz, M., Damen, D., & Mayol-Cuevas, W. (2018). Instance-level Object Recognition Using Deep Temporal Coherence. In Advances in Visual Computing: 13th International Symposium, ISVC 2018, Las Vegas, NV, USA, November 19 – 21, 2018, Proceedings (pp. 274-285). (Lecture Notes in Computer Science; Vol. 11241). Springer, Cham.* (Chapter 4).
- *Lagunes Fortiz, M., Damen, D., & Mayol-Cuevas, W. (2019). Learning Discriminative Embeddings for Object Recognition on-the-fly. IEEE International Conference*

on Robotics and Automation (ICRA) 2019, Montreal, Canada, May 19 – 21, 2019. (Chapter 5).

1.5 Thesis Outline

The outline of the thesis is as follows:

In Chapter 2, we provide a literature review on the topics of CNNs for object recognition, object detection, domain adaptation, domain generalization, few-shot learning, and incremental learning, which present similarities to the core challenges in the *in-situ* learning problem. These core challenges include having a model able to learn new instances, without finetuning the model, from a few sequential data, and generalizing to unseen environments. We discuss the metric learning approach (*a.k.a.* similarity learning) with more detail, due to its goal of learning discriminative features. We also discuss a related work for learning new objects without the need for retraining the model, and its methodology to evaluate the recognition performance of a model learning objects in such a regime.

In Chapter 3, we propose an approach to multi-view instance-level object recognition that combines convolutional and recurrent neural networks for exploiting the *temporal coherence* present in video data. We first evaluate sampling methodologies for using training data efficiently. We then present two temporal filters for exploiting the assumption that on video data the same object remains present over a number of consecutive frames. The temporal filters are a cumulative moving average and a machine learning approach using bidirectional long-short term memories. We evaluated these temporal filters on T-LESS and CORe50. We show how this way of learning can be especially useful when a few data points are used for training. Furthermore, using training data efficiently can accelerate the training process, which is useful for agents exploring the world in front of them. With our temporal filters, we achieve a precision of 99.13% for CORe50 and a mean average precision of 56.23% in T-LESS.

In Chapter 4, we address the problem of learning to recognize objects using example data from a specific environment and generalize to unseen ones, which is a core aspect in the learning scenario that we are after. The features learned by our model are discriminative between identities and aim to be indistinguishable between domains. Our approach proposes a supervised residual autoencoder (S-RAE) that combines image classification and reconstruction. We empirically evaluate our approach on eight object recognition datasets used for robotics and computer vision recognition tasks, and achieving an aver-

age 5.23% higher performance compared with the closest baseline and over 20% higher performance compared with a *state-of-the-art* adversarial network approach. Our proposed S-RAE however, assumes the availability of computational resources for learning new instances, which can be a significant limitation for current autonomous platforms. Therefore in the following chapter, we address the problem of learning generalizable feature of unseen objects, that can be used to learn to recognize new instances efficiently.

In Chapter 5, we address the problem of learning to recognize new objects without the need for retraining the backbone CNN. We propose a novel CNN architecture that learns to generate discriminative and separable features of an object's viewpoints by using a supervised triplet loss. To estimate an object's identity, we utilize a lightweight classifier in the features embedding space. The inference time in our framework lies in the order of milliseconds and can accommodate new instances easily. Our approach is easier to implement than current mining techniques, and the trained model can be applied to unseen objects. We evaluate our approach on four object recognition datasets used for robotics and computer vision applications: Amazon Robotics Challenge 2017 by MIT-Princeton, T-LESS, ToyBox, and CORe50 datasets. Our supervised triplets network, had a comparable performance to the finetuned models across all datasets, it achieves an average 4% higher performance for known objects, 5% for novel ones and 7% higher for the general case of combined known and novel objects, compared with the closest baseline.

In Chapter 6, we build on the findings from Chapters 3 and 5, for proposing a model that not only generates discriminative features but also considers the temporal consistency that must exist between neighboring video frames. We first replace the nearest neighbors search in the embeddings space for the nearest centroids algorithm for improving the inference time and scalability. We then propose a regularization term for reducing the distance between centroids and features from the same objects, and increase the separation between centroids and features from different objects. We then propose a modified version of the cumulative moving average filter, which depending on the distance between neighboring video frame features, it applies a weighted average that is particularly useful to quickly react to new objects on the scene. Overall, for the case of known objects, our model achieved an average 2.01% higher performance than the closest baseline, the S-Triplet network from Chapter 5. For the case of novel objects, it achieved a slight increase of 0.61%, an average of 3.21% higher recognition performance for the general case.

In Chapter 7, we conclude by discussing open challenges for the *in-situ* learning setup

and how this methodology has the potential to be applied to the related problems of object localization and segmentation.

CHAPTER 2

LITERATURE REVIEW

2.1 Related Work

This chapter presents a comprehensive literature review on the topics of object recognition, on-line learning, domain generalization, and few-shot learning, which are related to the idea of *in-situ learning*. We provide a short critical discussion at the end of each topic and its application to in-situ learning.

Handcrafted Feature methods for Object Recognition

Instance-level object recognition, *i.e.*, learning *specific* objects as opposed to object *classes*, was broadly studied during the past decades through the process of extracting and matching visual descriptors as in (Lowe, 2004; Wang et al., 2005; Bay et al., 2008; Damen et al., 2012; Hodaň et al., 2015). While most handcrafted approaches offer the benefit of real-time learning and inference, the design of a generic set of features and descriptors that can be used to any object is a difficult task (Ji et al., 2013), making them a challenging alternative for the *in-situ* learning of generic objects. We refer to generic object to indicate we do not make any assumption about the visual attributes of the objects, this is, if they are texture-less, transparent, with salient attributes, deformable geometry, etc.

On the other hand, machine learning approaches, such as the ones using convolution neural networks (Simonyan and Zisserman, 2015; He et al., 2016; Szegedy et al., 2016a),

offer the benefit of end-to-end learning of features directly from data, at expenses of typically requiring high amounts of it.

While there are approaches that combine hand-crafted features with CNNs (Lan et al., 2016; Kashif et al., 2016; Nanni et al., 2017), we prefer the end-to-end nature of deep learning models, considering their relatively straightforward implementation in robotic platforms, as oppose to a combination of hand-crafted with machine learning approaches which might require a more cumbersome integration. Furthermore, CNNs can be used to learn class-concepts or instance-concepts by providing an adequate label to be used by the cross-entropy loss. Therefore, the rest of the literature review, as well as the remaining chapters, focuses on the use of deep learning models for learning features that are used for recognizing generic objects.

CNNs for Object Recognition

Convolutional neural networks (*a.k.a.* ConvNets, CNNs, DNNs) domain as the current approach for object recognition and detection (Huang et al., 2017b). CNNs made their reappearance in 2012 when AlexNet (Krizhevsky et al., 2012) won the ImageNet large scale visual recognition competition (ILSVRC-2012) by a significant margin. AlexNet architecture builds on LeNet, proposing ReLUs for learning non-linearities, data augmentation with dropout for avoiding over-fitting and trained on consumer GPUs. The following year, ZFNet (Zeiler and Fergus, 2014) won ILSVRC, its architechture it is based on AlexNet but proposing smaller kernel size, going from 11×11 to 7×7. At same year VGG (Simonyan and Zisserman, 2015) was presented, proposing even a smaller 3×3 kernel with stride and padding 1, followed by 2×2 max pooling. Small 3×3 kernel became the standard later on. The winner of ILSVRC 2014 was GoogLeNet (*a.k.a.* Inception Net) (Szegedy et al., 2015a), proposing multiple parallel convolutions and stacking layers with different kernel sizes, named *inception modules* for constructing the feature maps. On 2016, ResNet (He et al., 2016) was the winner model for ILSVRC, proposing residual connections between deep layers to alleviate the *gradient vanishing* problem on deeper models, proposing a CNN with 200 stacked layers. DenseNet (Huang et al., 2017a) and (Xie et al., 2017) are extensions of the skipping-connection mechanism by connecting blocks of layers instead of just individual layers. GoogLeNet has updated accordingly to this skipping connection and batch normalization (Ioffe and Szegedy, 2015) which has proved to be useful to the optimization occurring underneath (Szegedy et al., 2016b), (Szegedy et al., 2016a). Additionally, Fully Convolutional Networks (FCN) (Shelhamer et al., 2015) and All Convolutional Net (Springenberg et al., 2015) propose using only

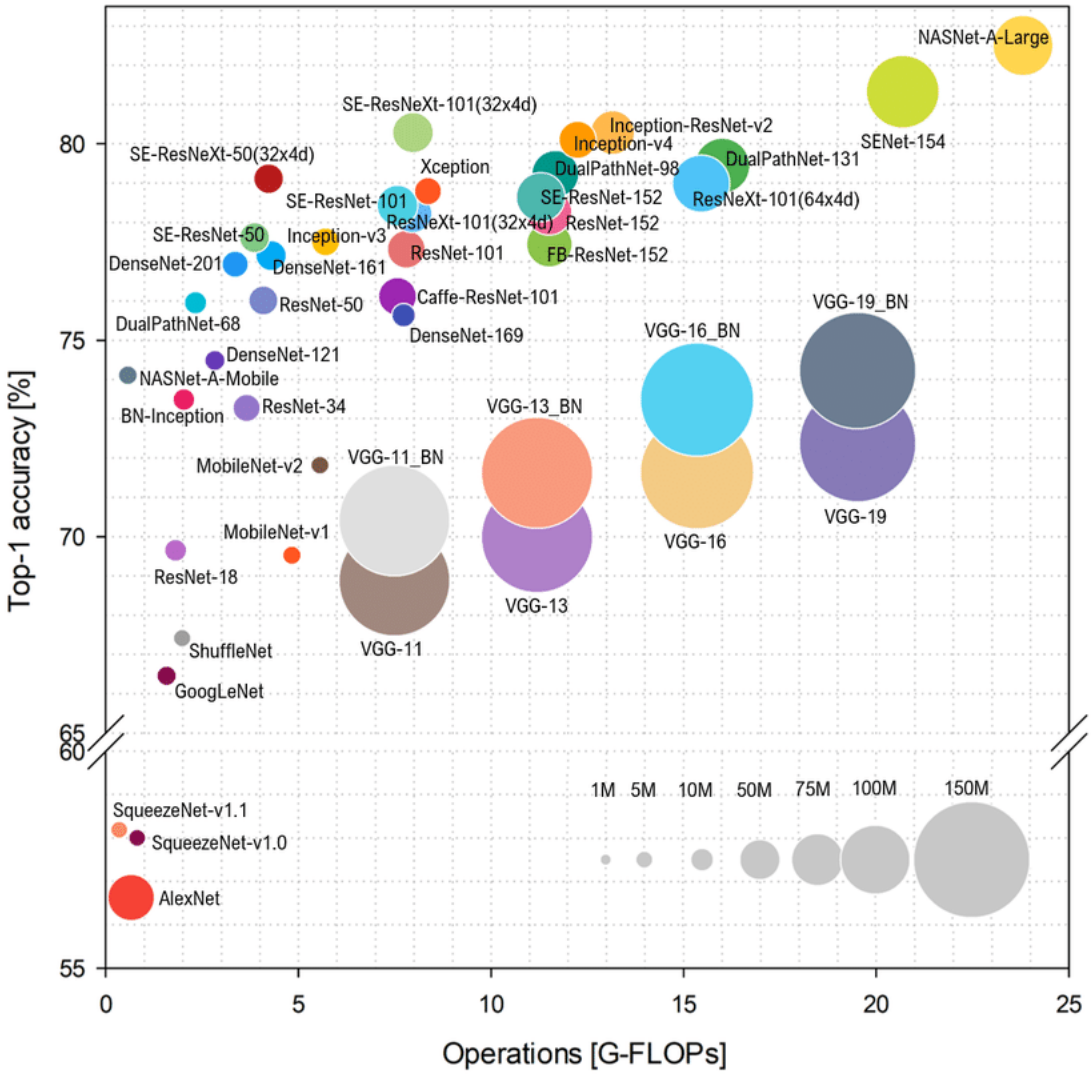


Figure 2.1: Comparison of model complexity (measured in FLOPS) of *state-of-the-art* architectures for the ImageNet dataset. Image from (Bianco et al., 2018).

convolutional layers and remove the costly fully connected ones, which now is done on most recent models.

Consistently, deeper architectures lead to higher precision on large-scale web-gathered datasets like ImageNet (Russakovsky et al., 2015) and PASCAL (Everingham et al., 2010), as shown in Figure 2.1, but it was unclear if such deeper architectures were essential for less complex recognition tasks such objects in CIFAR-10 dataset, or a few objects needed to be learned in-situ. Pioneering work on this matter is (Ba and Caruana, 2014), within which they show that a shallow architecture (*student*) can mimic complex functions learned by deeper models (*teacher*) on the CIFAR-10 and TIMIT datasets (an

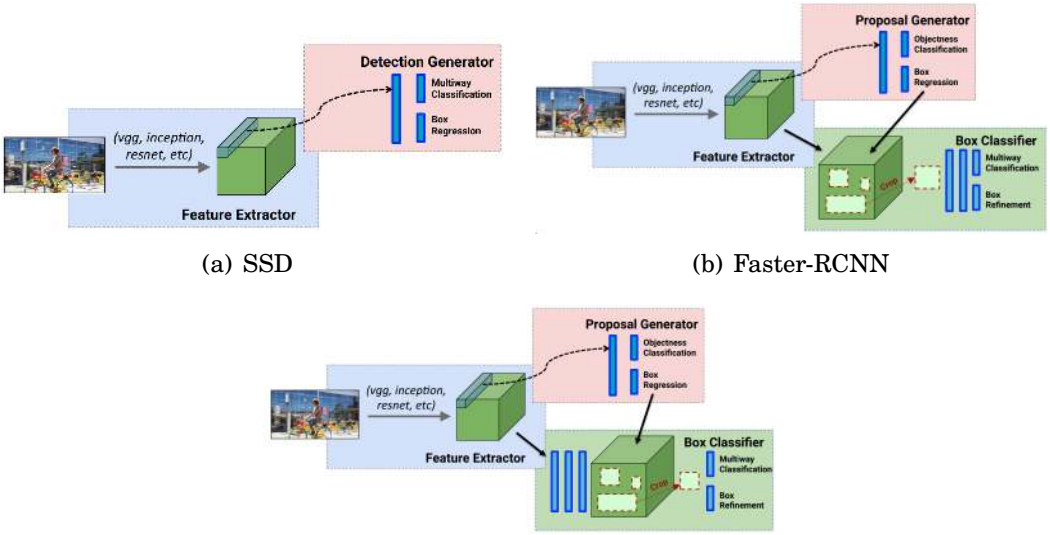
audio dataset). However, (Urban et al., 2017) empirically demonstrates that *distillation* (Hinton et al., 2015) only works if the shallow model contains a sufficient number of convolutional layers to learn functions comparable to deeper models and also demonstrating that CNNs do need to be both deep and convolutional for solving complex recognition tasks. These results are consistent with similar experiments performed by (Chatfield et al., 2014). Another empirical studies related to CNNs architecture are (Jarrett et al., 2009) and (Poggio et al., 2017), which focuses on the usefulness of the non-linearities (such as ReLUs) applied after the filter banks, as well as the effect of stacking layer to form deeper architectures. Results showed that such non-linearities and performing local contrast normalization are the most important ingredients for achieving competitive accuracy on object recognition benchmarks and that deeper architectures are capable of achieving a higher generalization than their shallow counterparts.

Designing architectures is a process with a significant amount of engineering, e.g., deciding kernels sizes, layers to stack, depth of convolutional maps, etc. Approaches such as NAS-Net (Zoph et al., 2018), Slim-Net (Liu et al., 2017) and Efficient-Net (Tan and Le, 2019) uses an *agent* for automating such task and learning a sufficient architecture for the dataset of interest, aiming for a more efficient model's capacity. Nevertheless, the task of learning an optimal architecture for a given task is still an open problem (Wistuba et al., 2019). Interestingly, how convolutional filters are connected seems to be a non-relevant aspect, since deep architectures with random connections are capable of achieving *state-of-the-art* recognition performance, as the ones propose in (Xie et al., 2019).

The first take on CNNs is that, although they generally achieve a higher recognition performance than hand-crafted features, for large-scale datasets, the complexity on their architectures and the learning process by gradient descend, make them a challenging alternative for autonomous agents that need to learn and react as soon as possible with constraint resources on-board.

CNNs for Object Detection

The use of CNNs for object detection builds on the aforementioned object recognition models used for learning features maps. For detection, additional layers are added for predicting the location, scale, and class of bounding boxes, or masks. While in this thesis, we focus on object recognition, the following literature review for object detection helps to see an immediate future direction of the propose solutions in Chapters 5 and 6.



(a) SSD

(b) Faster-RCNN

(c) R-FCN

Figure 2.2: Object detection approaches, consisting of CNNs with shared (a) and separated tasks for bounding boxes regression and image classification, (b) and (c). Image from (Huang et al., 2017b).

Region-based CNN (R-CNN) (Girshick et al., 2014) is a first approach that combines a heuristic method for producing bounding boxes using selective search. These bounding boxes are cropped and warp to a fixed-size image and feed to AlexNet for extracting the features and classify its content. Overfeat (Sermanet et al., 2015) proposes an agnostic-class box proposal by making use of the feature maps from AlexNet and applying an exhaustive-pooling approach for achieving better alignment between boxes and objects. A variant of this approach is (Erhan et al., 2014), which proposes a saliency-inspired neural network model for producing agnostic-class bounding boxes.

Spatial Pyramid Pooling Net (SPPNet) (He et al., 2015) introduces the idea of computing feature maps only once (rather than feeding each RGB image candidates to a CNN) and proposes an adaptive-size pooling layer, making this approach faster by two order of magnitude compared to R-CNN, but still not fast enough for real-time applications.

Fast R-CNN (Girshick, 2015) adopts the core ideas from SPPNet about computing the feature maps only once and having a Spatial pyramid layer of fixed size named region of interest pooling (RoIPooling). You Only Look Once (YOLO) (Redmon et al., 2016) uses the idea from Multibox (Szegedy et al., 2015b) for region proposals for bounding boxes, demonstrating that CNNs performs better than heuristic methods for this task and producing both region proposals and object classification in a one-step architecture. Its model is formed by three output layers: a softmax layer for classification, a linear layer

for regression the bounding boxes and a sigmoid layer for region proposals confidence.

Faster R-CNN (Ren et al., 2015) replaces the heuristic region proposal by Region Proposal Network (RPN) which is a small three-layers CNN and combining losses for box regression, box confidence scoring and object classification into one general loss function. SSD (Liu et al., 2016a) proposes a one-step (also referred as a *one-shot*) detection model such as YOLO, but considering feature layers with different scales and aspect ratios, outperforming YOLO in accuracy and speed.

Region-based fully convolutional networks (R-FCN), borrows the region proposal network from fast/faster R-CNN, but instead of cropping features from the same layer where regions proposals are predicted, crops are taken from the last layer, and thus sharing almost all the computations on the entered image.

YOLO was adapted to YOLO2 by following the approach of maximizing the shared computations on the image and keeping the one-shot architecture(Redmon and Farhadi, 2016). A step beyond bounding boxes are: Mask R-CNN (He et al., 2017), which builds on faster R-CNN for achieving pixel-wise segmentation by adding a parallel branch for predicting object-masks and SSD approaches (Poirson et al., 2016), (Kehl et al., 2017), (Mousavian et al., 2017) for 3D pose estimation.

Whether the bounding boxes are produced in a one or two-steps architecture, a trade-off between accuracy, running time, and memory requirements is present as empirically studied in (Huang et al., 2017b). While two-step approaches offer the highest accuracy, they also require more memory and computational time. Focal loss (Lin et al., 2018) is a first work for reducing the gap in performance between these approaches by proposing an objective function focused on the hard examples and reducing the negative effect from class imbalance. The selection of the image classification CNNs is part of this trade-off. Noticeable, sharing computations between the localization and recognition tasks had improved both accuracy and running times, but a large amount of training data is still required for achieving good performance. The lack of training data is typically addressed either by using large-scale datasets (Zeng et al., 2017) or data augmentation as in (Kehl et al., 2017).

Object detection is a closely related task to our problem that deals with the localization of objects on the image. The end-to-end nature of CNNs allows models that can perform both object detection and object recognition with the same CNN. Furthermore, although object detection is out of the scope in this thesis, it is possible to assume that our following contributions can be integrated into an end-to-end in-situ object detection or segmentation pipelines.

CNNs for Object Recognition From Video Data

Since both training and testing data come from video data in the *in-situ* setup, we now review the related work on object recognition from video data.

Due to the success of CNNs for learning visual representations from static images, applied them to video data was a natural progression (Karpathy et al., 2014). Using 3D Convolutions such in (Tran et al., 2014) was a pioneer approach for object classification from video data, by using a 3D convolutional kernel over video frames on the Intel's egocentric recognition of handled objects (Ren and Philipose, 2009). The use of 3D convolutions on video data became useful for video classification and action recognition such in (Hara et al., 2017) and (Wang et al., 2018), and for crowded prediction flow like in (Zhang et al., 2017b), with the limitations of higher memory requirements for processing longer sequences and the need of large-scale datasets for avoid overfitting.

Fusion methods emerged for the tasks of action recognition and video classification. Initially, raw-RGB frames were fused across the video sequence (fused in the form of averaging, concatenation, convolutions, pooling, correlation, or point-wise multiplication (Karpathy et al., 2014)). Since *motion* plays an important role for video and action recognition, a two-stream approach is proposed in (Simonyan and Zisserman, 2014a). In two-stream, an RGB and optical-flow streams are feed into similar or even sibling networks, and they are fused at some point in the network into one final tensor as in (Simonyan and Zisserman, 2014b) and (Feichtenhofer et al., 2017a).

An extension of 3D convolutions are CNNs with recurrent connections, such T-ResNet (Feichtenhofer et al., 2017b) and recurrent ResNet (Kemaev et al., 2018), which achieve a similar performance than two-stream approaches but only with RGB frames. While recurrent ResNet shares the weights among all frames, keeping the network size small, rather than incorporating temporal information by extending the convolutions over temporal windows as in T-ResNet, they both suffer from high memory requirements for processing long sequences. Nevertheless, it has been reported that recurrent connections and 3D convolutions are able of capturing motion information occurring within the video sequence, and thus they became more popular for video and action classification rather than in object classification.

To address the problem of high memory requirements for long video sequence for video classification, (Ng et al., 2015) proposes combining 2D CNNs for extracting feature maps and Long-Short Term Memories (LSTMs) for processing the sequential information. Since the variables are shared across the hidden states on LSTMs, this approach can handle 120 frames during training as opposed to 30 frames on the 3D CNN approaches.

To consider the motion information, they employ raw frames and optical flow as in two-stream approaches. Since CNN-LSTM models can work with long sequences as input and variable size sequences as output, they became useful for video description (Donahue et al., 2017), 6-D camera re-localization (Clark et al., 2017) and object tracking (Tripathi et al., 2016), (Ning et al., 2017). Variations of this approach are VideoLSTM which builds on the CNN-LSTM architecture but proposing an attentional model based on motion within the LSTM's hidden states (Li et al., 2016) and Convolutional LSTMs (ConvLSTM) which replaces the fully connections on the hidden states for 2D Convolutions (Shi et al., 2015) and aiming to capture more temporal relations across sequences. While the combination of CNNs with LSTM offer a solutions for processing long video sequences, how to efficient jointly training them is still an open problem (Donahue et al., 2017) and CNNs with additional LSTMs are commonly trained on two separated stages.

Top-down feedback and lateral connections as in CortexNet (Canziani and Culurciello, 2017) is another approach, where a pair of Convolutional and Decovolutional towers are interconnected, rather than considering multiple frames at the time like in 3D ConveNets, but processing only current and previous frames. Finally, learning features from moving objects in a semi-supervised manner as in (Pathak et al., 2017) is another alternative for exploiting video information and learn visual representations.

Including temporal information into a spatial visual model aids to increase the robustness against nuisances. However, state-of-the-art approaches relies on cumbersome 3D convolutions and recurrent connections that highly increases the redundancy in CNNs, making them not suitable for in-situ learning.

CNNs Limitations

Regardless of their biological inspiration and even surpassing Human-level performance on visual recognition challenges such as ILSVR, CNNs are still far from human-level learning capabilities (Lake et al., 2017). Arguably, the most significant limitation is that CNNs does not learn the semantics of training data but only an input-to-label mapping (Hosseini and Poovendran, 2017; Zhang et al., 2017a; Zhou et al., 2016). This input-to-label mapping can be fairly discontinuous to a significant extent and imperceptible perturbation can lead to misclassification (Szegedy et al., 2013; Fawzi et al., 2016) or even one-pixel perceptible perturbations can cause the same effect (Su et al., 2017). Capsule Networks (Sabour et al., 2017), Network In Network (Lin et al., 2013) and Graph CNN (Bruna et al., 2014; Schlichtkrull et al., 2018), are approaches for overcoming this limitation by using a sub-network (such a Multilayer Perceptron, a Capsule or a Graph)

as an alternative to purely convolutional kernels and aiming to learn from data in a more meaningful way.

Another aspect is their limited invariance to translation, mainly achieved by the pooling operations, and the even smaller invariance to rotation. Harmonic Networks (Worrall et al., 2017) and Rotation-invariant CNN (RICNN) (Cheng et al., 2016) are first approaches for addressing this limitation. Harmonic Networks replaces regular convolutional filters with circular harmonics, and RICNN proposes a regularization constraint on the objective function plus data augmentation. Based in RICNN, a common practice is just performing data augmentation in the form of random rotations on the training examples at expenses of more epochs required for convergence, but with the benefit of using the CNN filters (often from pre-trained models) without any modification.

Computational resources are another significant limitation, especially for robotic and mobile platforms without dedicated GPUs. Mobilenets (Howard et al., 2017), a fully Convolutional architecture with depth-wise separable convolutions and XNOR-Net (Rastegari et al., 2017), a CNN with binary weights (as opposed to floating-point values), are two models able to be deployed on devices with limited computational resources. Nevertheless, they still need to be trained on dedicated (and more powerful) platforms.

Finally, the amount of labeled data for avoiding overfitting is another critical limitation for deep architectures. Transfer learning as in Learning without Forgetting (LwF) (Li and Hoiem, 2016) and the immensely popular approach of finetuning (Donahue et al., 2014; Yosinski et al., 2014; Pasquale et al., 2016a; Wagner et al., 2013; Furnari et al., 2017) and *continuous* finetuning (Käding et al., 2016) are current alternatives for learning from small datasets.

While recent work aims to design lightweight and more efficient CNNs, learning the parameters of neural nets with gradient descent implies that learning onboard will be challenging for autonomous platforms, as they need to learn and react within their environment quickly. Furthermore, current convolutional approaches struggle with changes in scale, occlusions, clutter, training from few-data, etc. Certainly, there is still a need for a more suitable way of learning generalizable features to be used by autonomous agents.

Recognition of Novel Objects *on-the-fly*

(Milan et al., 2017) and (Zeng et al., 2018) were the first approaches to deal with the challenge of learning new objects without finetuning the model, motivated by the Amazon Robotics Challenge 2017 (ARC-2017). During that competition, teams were asked to learn a set of 20 objects, provided only 45 minutes before the start of trials.

While most teams performed an improvised data collection and finetunned their deep learning models, (Milan et al., 2017) proposes a model that learns to generate generic masks to segment the objects in a box and a shelf. Once each object has been separated from the others, a metric learning approach is used to compare the similarity between a set of training images provided by Amazon which depicts the objects and the images collected by the Cartman robot.

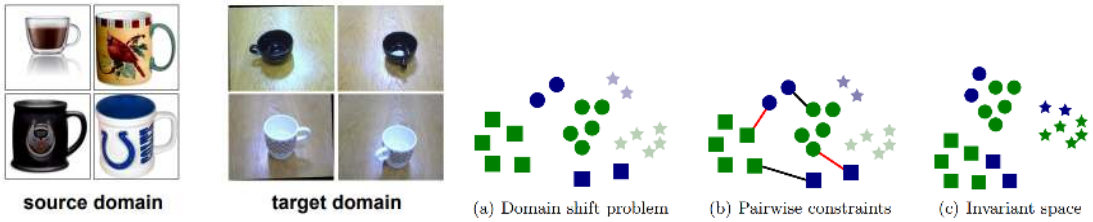
Similarly, the solution proposed by the MIT-Princeton team (Zeng et al., 2018), consists of a cross-domain image matching framework for recognizing grasped objects by matching them to product images, which uses a CNN architecture that learns novel objects without additional re-training. Their model consists of a two-stream CNN where one stream computes feature for product images (the images provided by Amazon), while the other stream computes feature vectors for the images captured by their robot. They optimize both streams so that features are more similar for images of the same object and dissimilar otherwise. During testing, product images of both known and novel objects are mapped onto a common feature space, where the nearest neighbors search takes places for associating embeddings from unseen objects. Additional details of these works are presented in the following section due to their closeness to our problem of learning novel object without finetuning the CNN.

Authors in (Zeng et al., 2018), offers a first approach to learning objects on-the-fly, by utilizing a discriminative model to learn features, and replacing the classification layer by the nearest neighbors search in the embeedings space. This is a core a idea that we refine in Chapters 5 and 6.

Domain Adaptation and Generalization

The problem of the differing train and test data distributions is present in the *in-situ* learning that we are after since the environment in which the training data was collected, might be different from the environment in which the given object is desired to be re-detected, as depicted in Figure 2.3.

A first approach that addresses this problem is domain adaptation. There are two variants within this approach: Unsupervised and semi-supervised domain adaptation. In the first variant, no target domain labels are provided during training, and only source domain labels are available. In the semi-supervised case, the model has access to domain labels and a few target domain labels (Chopra et al., 2013). In both cases, however, the model has access to data points from both source and target domains, a requirement that might be too expensive to obtain for robotics and autonomous systems applications,



(a) Dataset depicting differing training and testing conditions.

(b) Features Space

Figure 2.3: Domain shift problem, resulted from training and testing images with deferring distributions. The goal in this setup is to learn features which are invariant to the domains. Image from (Saenko et al., 2010).

which motivates our work on learning from a single source domain only. Within the aforementioned variants generative adversarial networks (GANs) (Liu et al., 2018b; Ganin et al., 2016; Long et al., 2018; Zhang et al., 2018; Duan et al., 2009; Pinheiro, 2018; Wulfmeier et al., 2017) posits as the *state-of-the-art* approaches for learning disentangled and transferable representations. The first limitation of these approaches is that require at least two training domains, to learn disentangled representations, which might not be feasible in the *in-situ* learning set up, where only training data from a single environment is available.

In domain generalization, there is no availability of target domains, but multiple source domains might be available (Li et al., 2018a). Within this approach lies the case of learning from a single source domain, which is the most related case to the *in-situ* learning that we are after. In this domain generalization variant, metric learning posits as a popular strategy for learning discriminative and generalizable features as in (Schroff et al., 2015; Wen et al., 2016; He et al., 2018; Wang et al., 2018; Li et al., 2018b).

There is a need for learning features invariant to domains, which is a core aspect for in-situ learning, most approaches assume that only known objects will be seen at deployment. Making re-training necessary to learn additional objects, and therefore challenging for in-situ learning.

Few-shot Learning

Few-shot learning is the task in which a model must be adapted to accommodate new objects not seen in training, given only a few examples of each of these novel classes, an extreme case of this set up is one-shot learning where a single example us used for learning each novel class. Few-shot is related to *in-situ* learning since the agent (either a

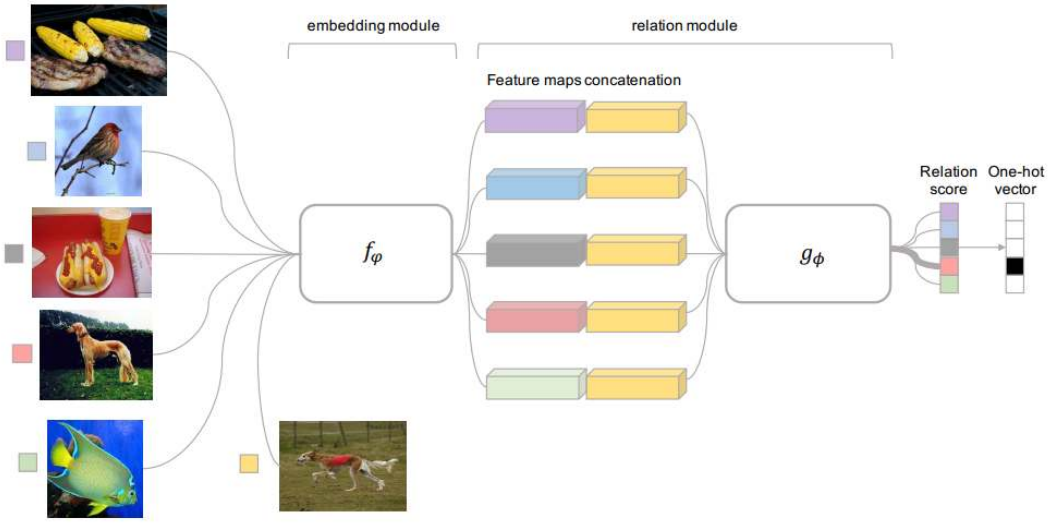


Figure 2.4: Approach for few-shot learning that consists of a trainable module that can select the closest class to a query image, building on the concept of image similarity. Image from (Sung et al., 2018).

human or an autonomous machine), might collect only a limited amount of training data, as oppose to collecting thousands of examples about the object desired to be learned.

The two main considerations in this set-up are: (a) the learning of the novel categories needs to be fast and (b) to not sacrifice any recognition accuracy on the initial categories that the CNN was initially trained on (Gidaris and Komodakis, 2018). The availability of only one or very few examples challenges the standard finetuning practice in deep learning (Pasquale et al., 2016b), while data augmentation and regularization techniques can alleviate over-fitting in such a limited-data regime, certainly they do not solve it (Pahde et al., 2018). Regarding finetuning the model, is still a slow procedure, requiring many weight updates using stochastic gradient descent, mostly due to the parametric aspect of the model, in which training examples need to be slowly learned by the model into its parameters.

A straightforward strategy for dealing with the lack of data from the new objects is to use a generative model to generate either synthetic examples (Bucher et al., 2017; Zhu et al., 2018) or synthetic features (Mishra et al., 2018; Schwartz et al., 2018; Xian et al., 2018), and then finetune the model by adding the synthetic data generated. While these approaches achieve promising results, the required finetuning step is a limitation in our *in-situ* learning problem.

Following the idea of comparing features from a query image against few-shot sam-

ples, in prototypical networks (Snell et al., 2017), authors proposes the idea that there exists an embedding (*a.k.a.* prototype, examplar) in which points cluster around it for each class, so the comparison step can be made more efficiently. To achieve this clustering, authors propose a model that learns a mapping of the input into an embedding space using a neural network and take a class's prototype to be the mean of its support set in the embedding space. Then, classification is performed for an embedded query point by simply finding the nearest class prototype. In (Ren et al., 2018), authors extend prototypical networks for semi-supervised learning where all unlabeled examples are assumed to belong to the same set of classes as the labeled examples of the episode, as well as the more challenging situation where examples from other distractor classes are also provided.

Matching networks (Vinyals et al., 2016) is another example of the idea of comparing features by proposing an attention mechanism over a learned embedding of the labeled set of examples (*a.k.a.* support set) to predict classes for the query point. Matching networks can be interpreted as a weighted nearest-neighbor classifier applied within the features space by a LSTM.

Another successful strategy is by utilizing meta-learning, where a CNN is trained to recognize a specific set of categories on a large-scale labeled dataset (e.g.,ImageNet), which provides a fairly generic bottom and middle layer units and then finetuning the top layers and including low-density separators (Wang and Hebert, 2016) or by utilizing a relational module that compares query images against few-shot labeled sample images (Sung et al., 2018), as illustrated in Figure 2.4. In both cases, the meta-learner need a significant corpus of data to achieve generalization. The metric learning approaches in Chapters 5 and 6 have the same limitation for learning generalizable representations.

The metric learning approaches in this thesis build on the same idea of comparing the features from a query image against a set of few-shot labeled examples. However, we propose utilizing a much simpler comparison mechanism, being Nearest Neighbors, as opposed to a more sophisticated trainable approach that might require a large corpus of data for achieving generalizability.

Few-shot learning approaches aim to learn class concepts from a few K-examples, as it might be the case in some extreme cases of in-situ learning of objects. Although state-of-the-art approaches assume the availability of other data modalities, such as text description, to assist with the learning from a few examples. However, approaches that use similarity learning, such as SimNet, seem a suitable strategy for in-situ learning, as potentially can be applied to unseen objects by replacing the classification layer by a

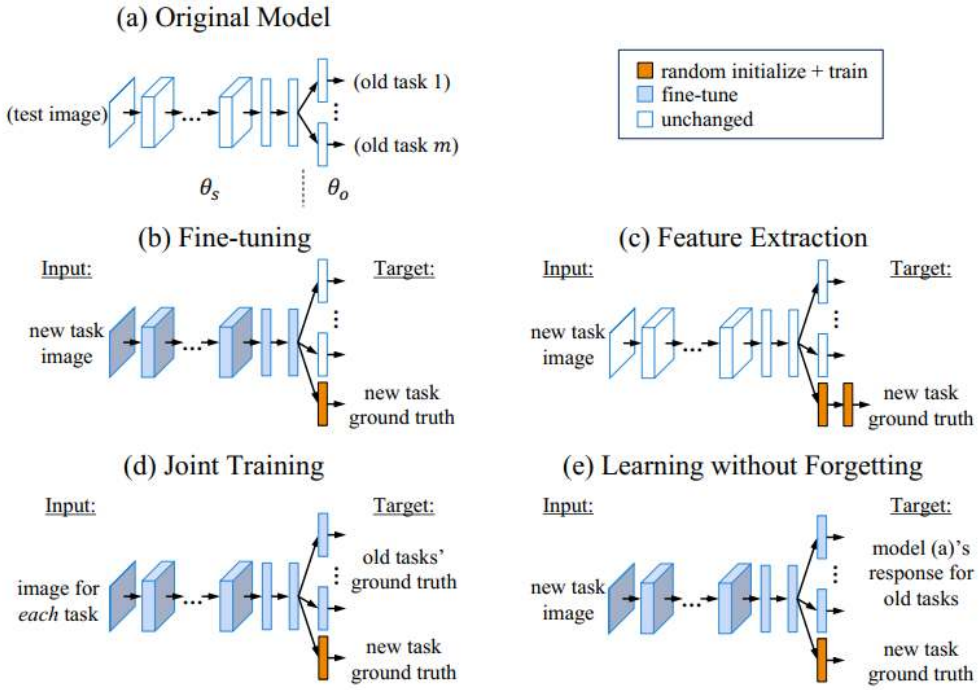


Figure 2.5: Learning without forgetting approach, consisting of model that can be re-trained for new classes without compromising the previously learned ones. Image from (Li and Hoiem, 2016).

similarity search in the features space. Furthermore, we do not consider in-situ learning with the case of very few training examples per object (usually one, or five, in this setup), as autonomous agents and demonstrator can easily collect hundreds of examples from an object, and leave this challenging scenario as a future research direction.

Incremental Learning

Also known as continual and lifelong learning, incremental learning has to do with exploiting learned information to rapidly store new memories without forgetting old ones, attribute that it is present in the *in-situ* learning in this thesis.

As mentioned in the few-shot learning section, in the flied of neural networks, the most common way of learning new objects is to finetune the network. However, as features relevant to the new task are learned through modification of the network's weights, which are relevant for prior tasks might be altered, leading to deterioration in performance referred to as *catastrophic forgetting* (French, 1999), Figure 2.5. Without access to older training data due to the lack of storage space, data rights, or deployed

nature of the agent, which are all very realistic constraints, naive finetuning is not a viable option for incremental learning.

One the other hand, *feature extraction*, where the model learned for the old task is re-used to extract features from the new data without adapting the model parameters, is neither a desirable strategy since the model tends to be highly conservative for the old objects and sub-optimal for the new ones (Triki Rannen et al., 2017).

Initial attempts to alleviate catastrophic forgetting relied on an episodic memory system that stores past data, like in (Rebuffi et al., 2017; Lopez-Paz and Ranzato, 2017). In particular, recorded examples are regularly replayed with real samples drawn from the new objects, and the network parameters are jointly optimized. A major drawback of the memory-based approach is that it requires large working memory to store and replay past inputs. Moreover, such data storage and replay may not be viable in some real-world situations (Farquhar and Gal, 2018).

A similar idea, but without requiring the actual original dataset, is the generative model named phantom sampling (Venkatesan et al., 2017), which generates hallucinated samples for re-training the model as if the original training set was available and supervision from the initialized network itself using a softmax loss. A similar approach is the performed by (Shin et al., 2017), where an adversarial network (GAN) mimics past data. Generated data are then paired with a corresponding response from the past task solver to represent old tasks. Called the scholar model, the generator-solver pair can produce fake data and desired target pairs as much as needed, and when presented with a new task, these produced pairs are interleaved with new data to update the generator and solver networks.

An alternative strategy is Progressive Neural Networks (Rusu et al., 2016), where the network architecture is replicated for every new dataset, with each new layer augmented with lateral connections to corresponding older layers. The weights of the new layers are then optimized while keeping the weights of the old layers frozen. The initial layers are thus unchanged, while the new ones can re-use representations from the older tasks. An unavoidable drawback of this approach is that the size of the full network keeps increasing with the number of added objects, which might be a limitation for today's autonomous platforms. Expert Gating (Aljundi et al., 2017), follows the same concept of a network of experts where objects (experts) are learned and added to the model sequentially, building on what was learned before and leading to the same drawbacks as progressive networks.

A more efficient approach is Learning without Forgetting (LwF), which preserves

responses of the network on older tasks by utilizing a distillation loss, where response targets are computed using data from the current dataset. Similarly, in (Castro et al., 2018) uses a combination of cross-entropy and distillation loss functions, where old training examples are stored in a memory component, which is akin to an exemplar set for maintaining a small set of samples corresponding to the old classes. As a result, this approach does not require the storage of older training data. However, this very strategy of using a distillation loss can cause issues if the data for the new task belongs to a distribution different from that of prior tasks. As more dissimilar objects are added to the network, the performance on the prior tasks degrades rapidly.

With the idea of protecting critical parameters (Zenke et al., 2017) from being modified during the learning of novel objects, authors propose measuring and tracking the *importance* at each synapse during training and representing the local contribution of each synapse to the change in the global loss. Therefore, when new objects are required to be learned, they consolidate the important synapses by preventing them from changing. Thus, learning novel objects is mediated primarily by synapses that were unimportant for past tasks and therefore avoiding catastrophic forgetting of these past objects. In (Seff et al., 2017), authors propose a regularizer term based on elastic weight consolidation (EWC), using a GAN, in (Nguyen et al., 2018), the authors utilize approximate bayesian inference for computing new weights. In (Triki Rannen et al., 2017), an undercomplete autoencoder is trained after training the task model. The autoencoder captures the most important features for the initial set of objects, so when new objects are desired to be learned, this autoencoder is used to ensure the preservation of those important features. This is achieved by a loss on the reconstructions made by the autoencoder, restricting a subset of the features to be unchanged and give the model the freedom to adapt itself to the new task using the remaining capacity.

Utilizing non-relevant weights for learning novel objects is an attractive idea that aims to make more efficient use of a CNN, however, since re-training of the model is still necessary, it is not an applicable strategy for *in-situ* learning.

While state-of-the-art approaches focus on not catastrophically forgetting previously learned classes, they do assume the availability of high-end computational resources to re-train the model when new objects are learned. Similarly to the previsions cases, this assumption makes such approaches unsuitable for onboard learning and therefore, unsuitable for in-situ learning. However, these approaches highlight the importance of evaluating the performance in both seen and unseen objects, which we do in Chapters 4 - 6.

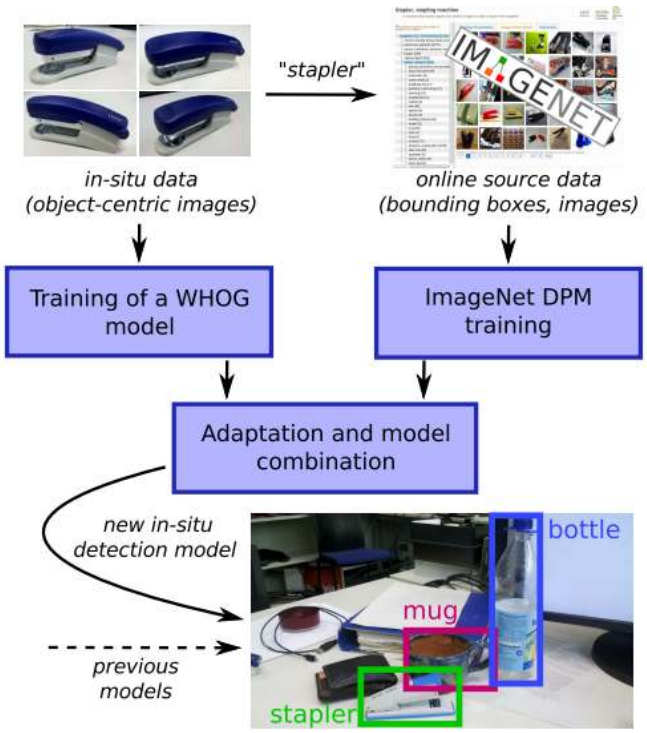


Figure 2.6: Approach for learning objects *on-the-fly*, consisting of an aggregation of features extracted from a large dataset (ImageNet) and features extracted *in-situ*. Image from (Göhring et al., 2014).

2.2 Critical Review on Related Approaches

(Göhring et al., 2014) is a pioneer work that addresses the problem of *in-situ* recognition of objects, understanding this as objects depicted within their natural or common environments, as opposed to pictures depicting them with ideal imaging conditions. The method consists of combining features from *in-vitro* datasets with features extracted *in-situ* to build a large-scale real-time recognition system (Figure 2.6). As limitations, this approach requires that the class of the desired object to be learned is present in the ImageNet dataset to build a robust classifier.

(Zeng et al., 2018) is a deep learning approach related to our *in-situ* learning problem since it addresses the challenge of learning novel objects without finetuning the model and from a few data. This work comes from the winner of the Amazon Robotics Challenge 2017 (ARC). The competition consisted of proposing an automated solution for stowing and picking common objects. A new challenge to the competition was dealing with unseen categories, provided only 45 minutes prior to the start of each task.

The proposed solution, showed in Figure 2.7, consisted of a two-branched CNN for

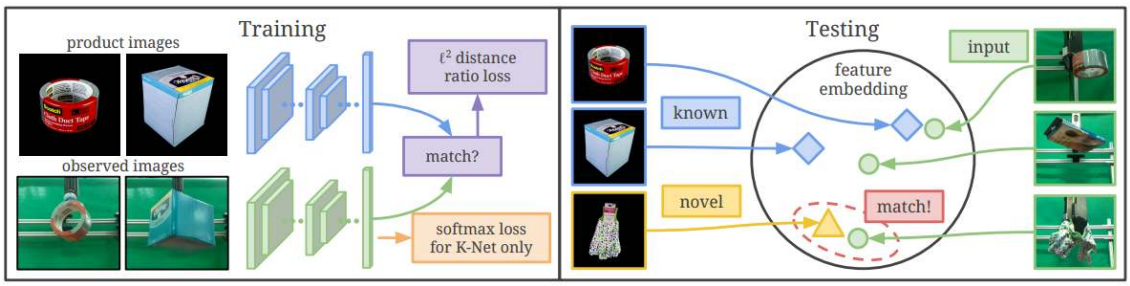


Figure 2.7: Supervised metric learning approach for learning object on-the-fly, consisting of a modification of the Siamese architecture for recognizing known and novel objects. Image extracted from (Zeng et al., 2018).

learning to associate product-images and images collected by their robot. The association is done by the nearest neighbors search in the features space and, as depicted in the right side of Figure 2.7, the goal was to have a discriminative model such that brings together features from the same instance and separated otherwise.

While this work was focused on matching product-images and images taken by a robot, authors empirically demonstrated that the nearest neighbor classifiers could directly be constructed in the feature space for learning novel objects. Furthermore, in Chapters 5 and 6, we pursue the same objective of having a model that produces embeddings, and we also utilize their propose methodology for evaluating the recognition performance when objects are learned *on-the-fly*.

The model in Figure 2.7 consists of a modification of the Siamese architecture (Koch et al., 2015), where each branch has a set of specialized weights for recognizing known objects, trained with a cross-entropy loss (K-Net), and the other branch trained to recognize novel objects, trained with a contrastive loss (N-net). Not only the model has to be trained in two different stages, but N-net is trained with examples chosen by a neural network which, according to the authors, leads to an increase in the recognition performance.

In Chapters 5 and 6, we propose a less cumbersome and easier to replicate framework, where a single-branch CNN is trained with a modification of the triplet loss (Schroff et al., 2015) and cross-entropy loss. Our approaches build on works like center loss (Wen et al., 2016) (Figure 2.8), where a single CNN can be trained with a combination of cross-entropy loss, to aid the optimization process and a metric learning loss, to make the features more discriminative.

Particularly, in Chapter 6 we focus on improving the inference speed and storage requirements by replacing the expensive nearest neighbors search for a more suitable

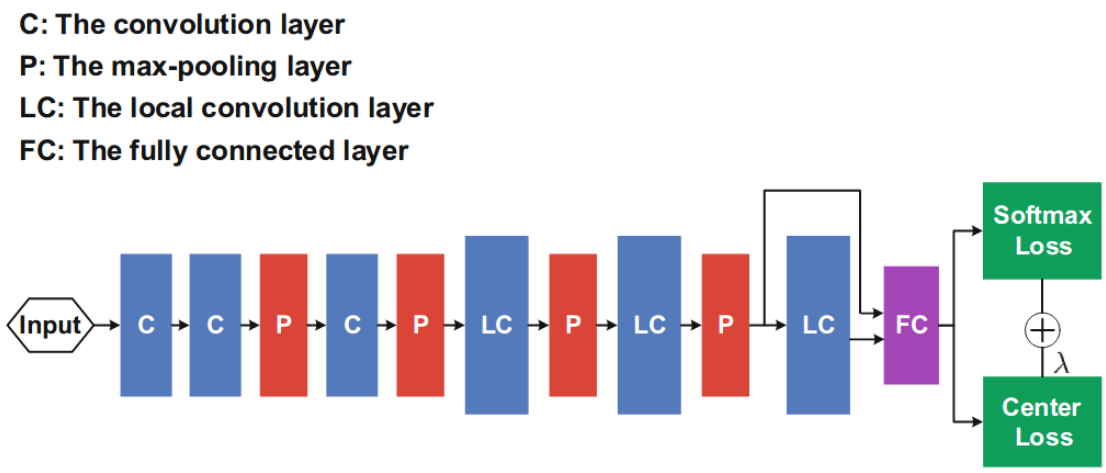


Figure 2.8: Center Loss approach for learning discriminative features, image from (Wen et al., 2016).

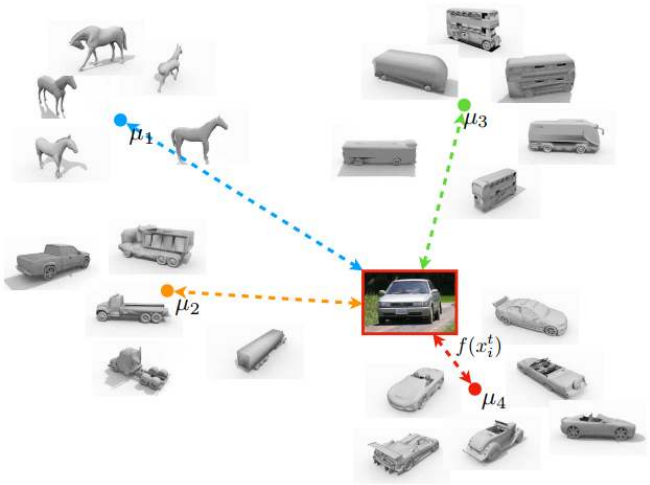


Figure 2.9: Instead of comparing a query image against all possible candidates, SimNet proposes comparing queries against class prototypes, taking advantage of discriminative prototypes generated by Similarity Learning approaches . Image from (Pinheiro, 2018).

nearest centroids search in the embeddings space. This improvement was inspired by (Pinheiro, 2018), were authors proposes a CNN that finds the most similar centroid (*a.k.a.*, prototype) to a given query image to achieve domain adaptation, as we show in Figure 2.9.

2.3 Object Recognition Datasets

In Table 2.1, we show contemporary public object recognition datasets to this thesis. We selected CORe50, Toybox, iCub-world, BOP, ARC-MIT, and In-situ Household for our experiments since they exhibit challenging conditions present in the *in-situ* learning problem, such as generalizing to different domains, dealing with changes in perspective, scale, motion blur, illumination, and clutter. We describe each dataset as follows:

CORe50: Originally proposed for continuous learning, this dataset shows 50 objects across eleven environments and allows us to test the generalization capabilities not only to unseen object's poses but also new environments. We utilize the standard testing set composed by scenes 3, 7, and 11. In Chapters 3 and 5, we use the remaining scenes for training. In Chapters 4 and 6, we consider the most difficult set-up of using only scene 1 for training the model.

Toybox: It is composed of 360 toys manipulated by a person. Toybox allowed us to evaluate how well the model scales, by learning 120 novel objects *on-the-fly*. We utilize the hodgepodge videos for training and the translations and rotations across x, y, z-axis for testing. The testing set also depicts new conditions such as changes in scale, partial views, and occlusions.

iCub-world: This dataset contains 200 household objects shown by a demonstrator to an iCub Robot. We selected the mixed manipulations set, taken with the left camera for training the model and mixed manipulations set but the following day for testing. The testing set depicts additional backgrounds, viewpoints, and scale.

BOP: It was initially proposed for 6D pose estimation and it is composed by six datasets (Linemod, Tejani, Rutgers-APC, T-LESS, TUD-L, and TOY-L), although some objects in this dataset were collected in-vitro, they offer the challenge of learning from few data with clean background images and generalize over clutter and occluded environments, which is the interest of this work.

ARC-MIT: This dataset was collected from the Amazon Robotics Challenge 2017, consists of 60 objects collected by the MIT-Princeton robot and contains training images from two different domains: Synthetic images provided by Amazon and real images captured by an RGB camera showing the objects being grasped by the robot. For this particular dataset, we use the methodology proposed by (Zeng et al., 2018) consisting of restricting the number of instances in the embeddings space. This is, every test image is compared against 20 possible instances, as opposed to the total of 60 instances, following the provided competition rules.

Table 2.1: Contemporary object recognition datasets.

Datasets	Objects	Description	Variations	Total Images
COIL-100 (Nene et al., 1996)	100	Household objects with a black background. RGB images collected with a turntable and a fixed camera	n/a	7,200
SOIL-47 (Koubaroulis et al., 2002)	47	Household objects with a black background. RGB images collected with a camera mounted in a robot arm	lighting conditions	1974
NORB (LeCun et al., 2004)	50	Texture-less and color-less toys. RGB images collected with a turntable and a fixed camera	jittered objects cluttered	97,200
ALOI (Geusebroek et al., 2005)	1000	Household objects with a black background. RGB images collected with a turntable and a fixed camera	viewing angles lighting conditions illumination color	110,250
3D Objects (Savarese and Li, 2007)	50	Household objects depicted within their common environment (a.k.a. in-situ). RGB-D videos taken by a user	scales viewing angles	7,000
Intel Egocentric (Ren and Philipose, 2009)	21	Household objects manipulated by a user. Train images collected with a turntable and test images with a wearable camera	clutter lighting conditions	70,000
BigBIRD (Singh et al., 2014)	125	Household objects. RGB-D images collected with a turntable.	N/A	60,000
RGB-D (Lai et al., 2011)	300	Household objects. RGB-D train images collected with a turntable and testing images collected in-situ by a mobile robot.	clutter occlusions lighting conditions	250,000
iCubWorld (Maiettini et al., 2017)	200	Household objects manipulated by a user, RGB images collected by the iCub's camera	scale clutter occlusions	200,000
ARC-MIT (Zeng et al., 2018)	61	Household objects manipulated by a robot, RGB images collected by the MIT-Princeton's robot	scale clutter occlusions	5,900
BOP (Hodan et al., 2018)	89	Household and industrial objects RGB images collected with a turntable	scale clutter occlusions	24,401
iLab-20M (Borji et al., 2016)	718	Toy vehicles within an emulated city, RGB images collected with a turntable	scales translations viewing angles	21,798,48
CORe50 (Lomonaco and Maltoni, 2017)	50	Household objects manipulated by a user, RGB images collected with a wearable camera	unseen backgrounds occlusions viewing angles lighting conditions	164,866
ToyBox (Wang et al., 2017c)	360	Toys manipulated by a user, RGB image collected with a wearable camera	scale translations rotations	2,300,000
In-situ household (ours, Chapter 6)	**20**	**Household objects with training and testing images collected in-situ. RGB images collected with a smartphone camera by a user.**	**unseen backgrounds lighting conditions clutter occlusions**	**12,000**

In-situ Household: Since none of the *state-of-the-art* datasets allow evaluating the more realistic situation where the training images are collected within the place where such items are commonly used, such as TV remotes in a living room, in Chapter 6 we propose a dataset is to assist the benchmarking of *in-situ* learning approaches by depicting each instance in its commonplace. The dataset is currently not publicly available, but we plan to make it available in January 2020, current status of the dataset can be consulted in (Lagunes-Fortiz, 2019b).

We do not use the remaining COIL-100, SOIL-47, NORB, ALOI, BigBIRD, and RGB-D datasets since they do not present enough level of nuisances to be comparable with the in-situ learning of objects, as shown in Appendix A. The intel-egocentric utilizes in-vitro images for generalizing to in-situ images, and we already use the ARC-MIT and BOP datasets to explore that case. The iLab-20M dataset was not available to download at the time of this thesis was submitted.

2.4 Summary

In this chapter, we have provided a literature review on the topics of CNNs for object recognition, domain adaptation, domain generalization, few-shot learning, and incremental learning. Although in these fields there is a need for learning generalizable features, most of the propose approaches assumes availability of high-end computational resources to re-train the model to learn new instances. Considering the challenging requirements in the *in-situ* learning of having a model able to learn new instances, without finetuning the model, from few sequential data and generalizing to unseen environments, we found the metric learning approach particularly useful given its goal of generating features which are close to each if they belong to the same object, even when the environments in which are depicted are very different and apart from each other if they belong to a different object. Therefore, the following proposed methods aim to produce discriminative features of seen and unseen objects. Additionally, in this chapter we identified the datasets: CORe50, Toybox, iCub-world, BOP, ARC-MIT, and In-situ Household, relevant for our experiments since they exhibit challenging conditions present in the *in-situ* learning problem, such as generalizing to different domains, dealing with changes in perspective, scale, motion blur, illumination, and clutter.

CHAPTER 3

INTEGRATING SPATIAL AND TEMPORAL FEATURES

In the previous chapter, we review state-of-the-art object recognition approaches based on CNNs and their limitations for being used to in-situ learning of objects. A first flaw is their temporal inconsistency while classifying object from video data caused by nuisances such occlusions, changes of perspective, ambiguous viewpoints, etc. Therefore, in this chapter, we start by proposing the use of temporal filters for addressing such lack of consistency. While the propose machine learning filter achieves state-of-the-art results, we find the cumulative moving average (CMA) algorithm to be of particular use for in-situ learning since it does not require a training stage, and it can be applied to unseen objects. Due to these properties, we revisit the CMA filter in Chapter 6.

3.1 Introduction

In this chapter we study the use of Convolutional Neural Networks (CNNs) for instance-level multi-view object recognition, where training data comes from a few seconds of video depicting the object of interest from a limited number of viewpoints and the testing data consists of a video sequence where instances appear at different times within the unseen and noisy video sequence, we illustrate this in Figure 3.1

This methodology for training and testing CNNs emulates sequential training data that an autonomous agent might have collected to learn to recognize a *specific* object and the testing sequence emulates an scenario where an autonomous agent sees an unknown number of learned objects. Our approach consists of exploiting the temporal coherence

(a) Training video sequence depicting Object #10 from the CORe50 dataset

(b) Testing video with multiple objects across the sequence from the CORe50 dataset

Figure 3.1: Training sequences with a single object and a testing sequence with all individual testing sequences concatenated.

present in video data, while the model can quickly react to a different object suddenly appearing in the video sequence.

Training CNNs for object recognition and detection is commonly achieved with class-level learning (Huang et al., 2017b). However, this approach of using large databases might be unsuitable to the widely encountered situation for intelligent agents performing tasks with *specific* objects around them. The need for a model that can be trained with a few data motivates our work, *i.e.*, that uses low hundreds of examples per instance. We also aim to achieve the same level of performance as one model trained with an order of magnitude more of training data.

First, we explore how the way of utilizing the video data influences the performance of a CNN for learning robust representations. We investigate if sampling frames by following a vertical-slices trajectory leads to the same level of performance of a more systematic way, such as using a sphere with all Point-of-Views (POVs) available. We also evaluate if frame-to-frame sampling leads to better results compared to normal-random sampling.

We then utilize a CNN for learning and extracting features and using temporal filters for exploiting the *temporal coherency* that is present on video data to increase the robustness against nuisances, such as, changes on perspective, illumination, scale object's pose, occlusions, and noise. We refer to *temporal coherency* to the assumption that on video data, the same object remains present over a number of consecutive frames,

meaning that abrupt changes in the predictions between neighboring video frames is unfeasible, as we illustrate in Figure 3.2.

We propose temporal filters which are applied to the frame-to-frame predictions from the CNN, for enforcing temporal coherence. This methodology differs from models with early recurrent connection as in *state-of-the-art* approaches such as CortexNet and Ladder Networks, which require more training data due to the increased number of recurrent connections.

The first filter that we propose is a simple algorithm that predicts the object's identity in a given frame by averaging the previous predictions over a definite number of frames. This method already offers an improvement but requires a careful selection of the number of frames to be considered, and avoid, as much as possible, fusing predictions containing different objects. The second filter is a recurrent neural network trained to produce a sequence of predictions with temporal coherence.

We evaluate our temporal filters with datasets used for instance-level object recognition: T-LESS, with 30 electrical and textureless objects and CORe50, containing 50 domestic objects. We show that models trained with a few view-points, but that exploit temporal coherence, models trained with a few data points perform with a similar performance to models that used the whole dataset. We use precision and mean average precision as the metrics for evaluating the recognition performance of each model.

3.2 Related Work

In instance-level object recognition, models learn *specific* objects as opposed to classes. This setup was broadly studied during the past decades through the process of extracting and matching descriptors as in (Lowe, 2004; Wang et al., 2005; Bay et al., 2008; Damen et al., 2012). This approach is not necessarily a flawed approach *per se*, but it can be challenging to integrate visual components. The effectiveness of end-to-end CNNs, specifically showcased on class-level object recognition, does not present such integration problem.

CNNs tend to propose deeper architectures for learning features at various levels of abstraction and achieve higher generalization capabilities (Goodfellow et al., 2016). While CNNs achieve lower error rates that hand-crafted featured methods for object recognition, they are still prone to errors on real-world applications considering a limited invariance to rotations (Cheng et al., 2016), occlusions (Osherov and Lindenbaum, 2017) and noise (Zheng et al., 2016; Fawzi et al., 2016).

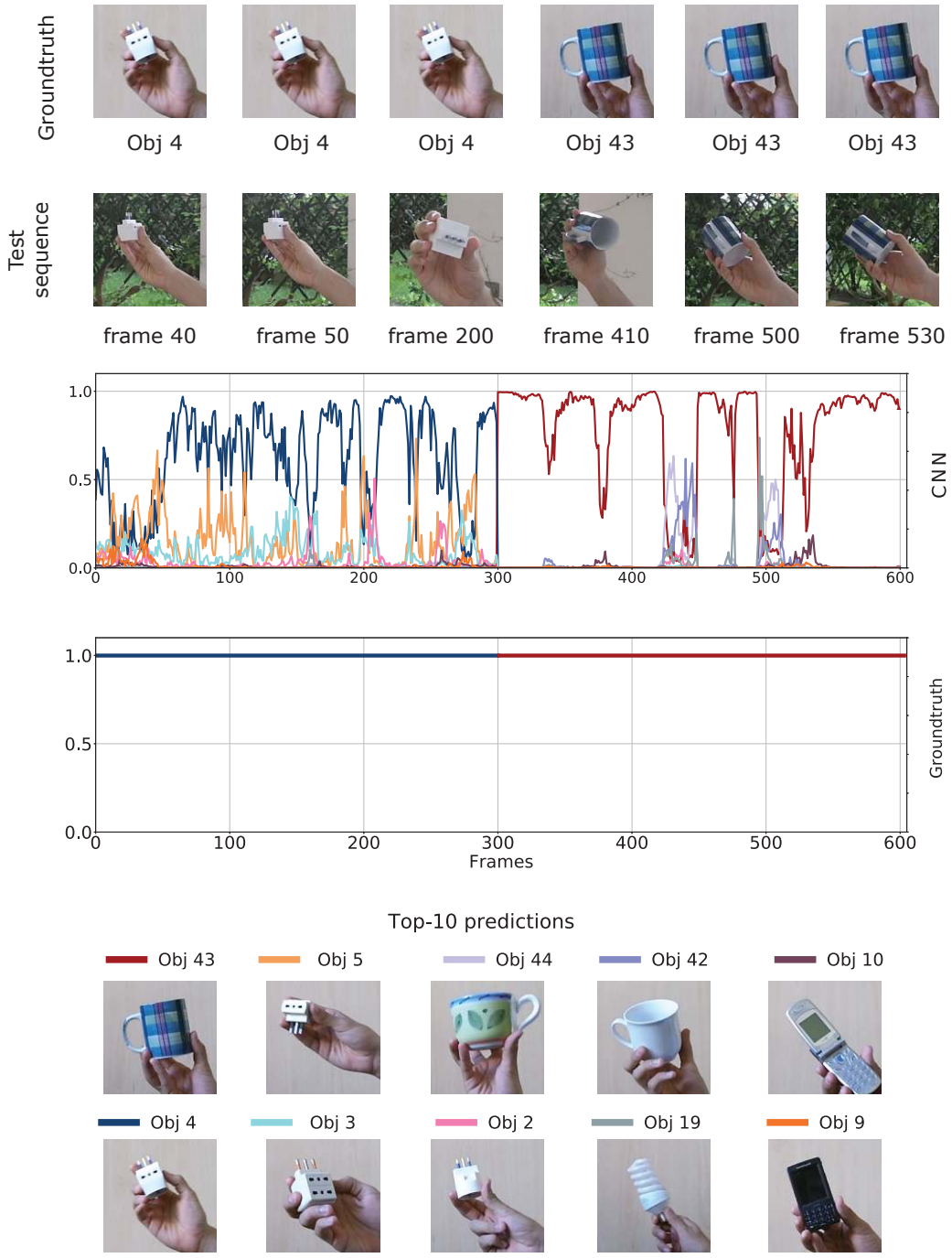
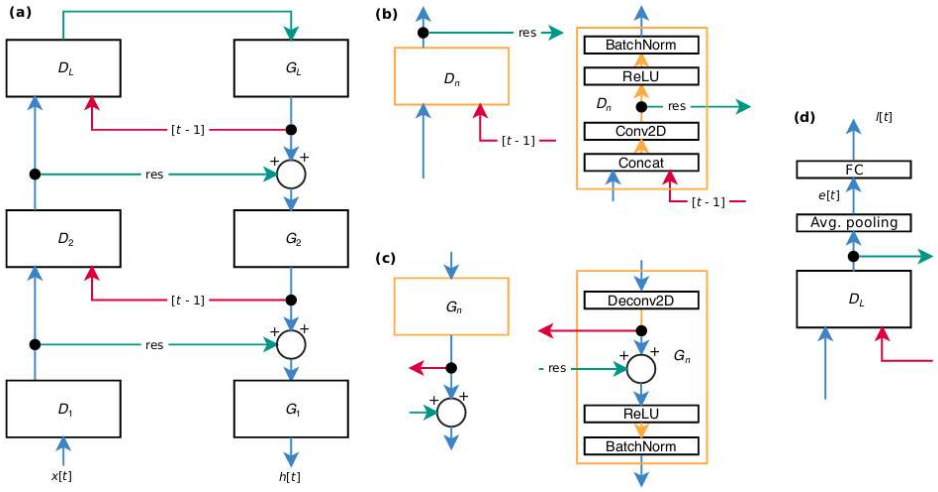
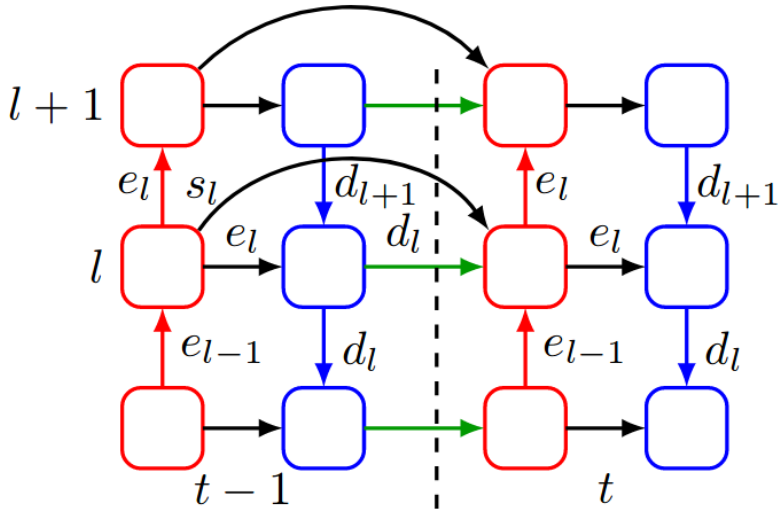


Figure 3.2: Temporal coherence. We present the top-10 predictions of a CNN trained to recognize the 50 instances in CORe50. We show how predictions are incoherent across the video sequence due to changes in the background, object's pose, illumination, etc.



(a) CortexNet architecture, image from (Canziani and Culurciello, 2017)



(b) Recurrent Ladder Network, image from (Prémont-Schwarz et al., 2017)

Figure 3.3: Models for combining temporal and spatial features.

CortexNet (Canziani and Culurciello, 2017) is a pioneer approach for addressing for exploiting the temporal consistency from video data. In CortexNet, the authors propose a model with feed-forward and lateral connections, inspired by the cortical areas of the visual system in humans. As depicted in Figure 3.3(a), the model architecture is composed of discriminative and generative blocks named D_n and G_n, respectively. The Generative blocks are fed with the superposition of top-down and bottom-up images and provide a modulatory input to its correspondent Discriminative block, based on the previous time step $t-1$ connections. A detailed representation of the discriminative and generative blocks are shown in Figure 3.3(a), where the authors show the branching and superposition operations happen right after the spatial projection onto the transposed convolutional kernels.

A more recent approach is Recurrent Ladder Networks (Prémont-Schwarz et al., 2017), which is an extension of the Ladder networks (Rasmus et al., 2015), consisting in a conducive to iterative inference and temporal modeling. This approach is similar to CortexNet by having the information first flowing from the bottom (the early layers in the CNN) to the top through a stack of encoder cells (discriminative blocks in CortexNet) and then, information flowing back from top to bottom using a stack of decoder cells (generative blocks in CortexNet). However, what makes this approach different from CortexNet is the way the encoder and decoder cells are using temporal information with horizontal connections. In Recurrent Ladder Networks, the discriminative blocks are connected between time t and $t-1$, while the generative blocks are connected sequentially with discriminative ones, as illustrated in Figure 3.3(b).

Recurrent Ladder Networks utilizes recurrent neural networks such LSTMs (Hochreiter and Schmidhuber, 1997) or GRUs (Chung et al., 2014) cells in the encoder and cells inspired by the original Ladder networks in the decoder. Interestingly, training the model is done by a weighted sum of the costs at different levels of abstraction. Where the highest abstraction level, like classification, is typically formulated at the latest layer. Conversely, the output of the decoder cell in the bottom level is used to formulate a low-level task, corresponding to abstractions close to the input. Low-level tasks can be denoising, object detection, segmentation, or predictions in a temporal setting. In Chapter 4, we show that training a CNN in a multitask setting, with tasks of different levels of abstractions, such as prediction and image generation, will allow the model to learn more discriminative features.

On the other hand, separating spatial and recurrent connection by having recurrent architectures, like LSTMs and GRUs, right before the logits prediction in the CNNs

has shown to be a useful approach for exploiting temporal information on video data for tasks such as 6-D camera re-localization (Clark et al., 2017) and object tracking (Tripathi et al., 2016). These approaches allows to process temporal information from more than a previous $t-1$ frame at expenses of having to trained the spatial and recurrent representations at different stages.

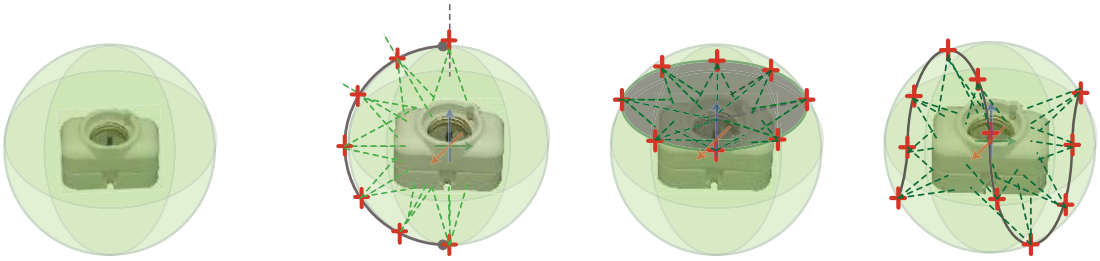
Our approach falls in the category of separated spatial and recurrent representations, motivated by the benefit of processing temporal information from multiple frames and not just the previous temporal frame. Our model operates in a less restrictive domain compared to existing work on data association like *object tracking*, where the location and identity of an objects are known in the first frame and the tracker predicts its localization on the subsequent frames. In contrast, there is no prior information to be required. Furthermore, it is different from *video classification* since we do not aim to model the motion occurring between video frames, in contrast with methodologies used on video and action recognition where information from middle and early layers is used by 3D convolutions (Ji et al., 2013) or recurrent connections (Hara et al., 2017).

3.3 Proposed Method

In order to evaluate how the way of utilizing the video data influences the performance of a CNN, with the T-LESS dataset, we first consider the case of an object's full viewpoints sphere, as we shown in Figure 3.4(a). We then evaluate exploration trajectories: A unique vertical slice from the sphere, as illustrated in Figure 3.4(b), a single horizontal slice taken at 45° respect to the horizon, as depicted in Figure 3.4(c) and a sinusoidal path that travels around the sphere, and can be seen as a combination of vertical and horizontal paths, as showed in Figure 3.4(d). Each of the paths generate a training set containing the **10%** of images compared to the full viewpoints sphere. Similarly, with the CORe50 dataset, we randomly sampled the video frames sequences to form training sets with a total of **10%** and **50%** images. We considered two approaches for sampling the video frames, being frame-to-frame and normal-sampling.

3.3.1 Architecture Design

Our approach consists of a combination of a *state-of-the-art* CNN, that produces features in the image space, with an additional temporal filter, that enforces the temporal coherence on the predictions generated by backbone CNN. The models are connected at the



(a) Full Sphere (b) Vertical Slices (c) Circular Slices (d) Sinusoidal

Figure 3.4: Proposed trajectories to form a training set with a total of 10% images compared to a full sphere (left) of Points of View.

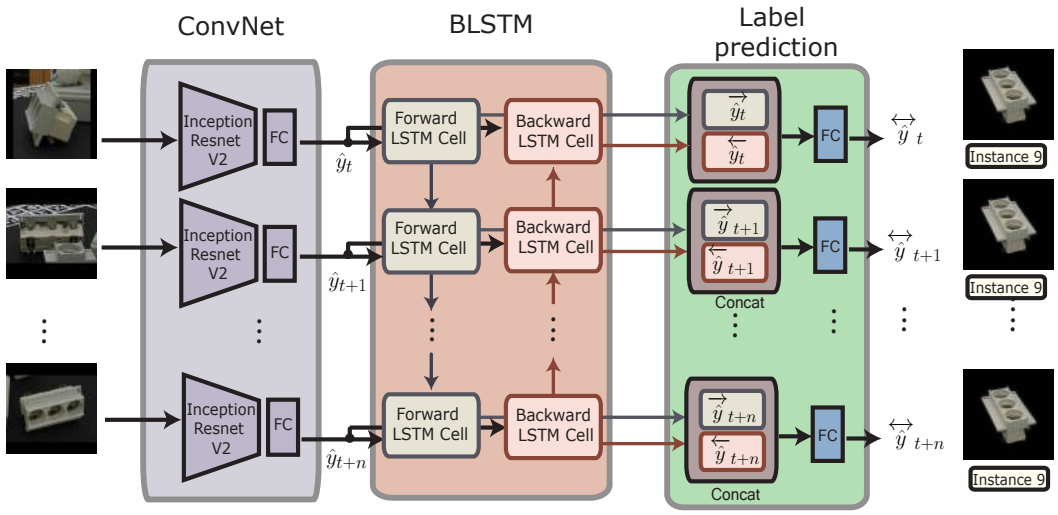


Figure 3.5: A continuous stream of images goes from the CNN and BLSTM for producing temporal coherent predictions.

last dense layer in the backbone CNN, which predicts the labels $\hat{y}$ at each time step.

Separating the backbone and temporal filters allow us to efficiently utilize a pre-trained CNN to transfer learning from a large-scale dataset such as ImageNet (Russakovsky et al., 2015) into our datasets and improve the over-fitting caused by the few training examples available. Additionally, we can efficiently increase the size of the video frame sequence, since the BLSTM utilizes low dimensional data as input, as oppose to having recurring connections across all the model and requiring high memory resources to compute the gradients. The disadvantage of this approach is that each network has to be trained in a separated stages.

The reason for requiring separated training stages is that feed-forward CNNs are trained more efficiently when the training examples are shuffled in the mini-batch

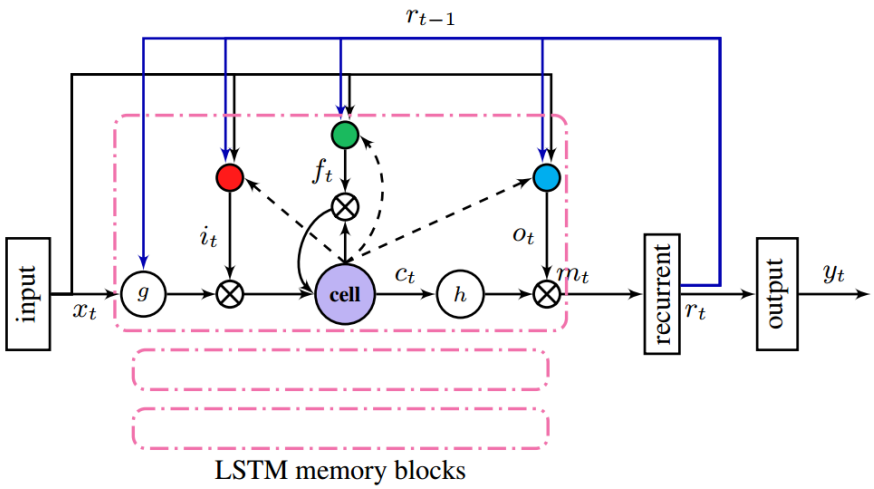


Figure 3.6: LSTM cell (hidden state h_t), with trainable input i_t, forget f_t, and output o_t gates for producing temporal coherent predictions. Image from (Sak et al., 2014).

(Goodfellow et al., 2016), in the other hand, BLSTMs require sequential training data for finding the short and long term features.

For training our framework, we first initialize our backbone CNN with a pre-trained model with the ImageNet dataset. We fixed the weights of the convolutional layers, and only the last fully connected layer is trained. We then trained all the parameters in the model using stochastic gradient descent, with the same training instances and labels as before.

The Bidirectional Long-Short Memory (BLSTM) model consists of a series of blocks (named *hidden states* h_t), as the one depicted in Figure 3.6. The input i_t, forget f_t, and output o_t gates, process the temporal information using the current image, and the output from a previous hidden state. What makes the BLSTM unique to other recurrent approaches, is that the model processes the data sequence in both forward (past to future) and backward (future to past) ordering, as shown in Figure 3.5. The advantage of using both directions, is that a current frame, can take into account information from future frames, in Figure 3.8, we present an example where the test images contain a high amount of clutter, and by utilizing the BLSTM filter, it is possible to utilize more discriminative information from subsequent frames.

To train the BLSTM, we freeze the variables from the CNN. The input i, forget f and output o gates are followed by ReLUs as non-linear activation function for each state h, as we shown in Equation 3.1. The weights W and bias terms b are shared across all the LSTM hidden states.

$$
\begin{aligned}
i_t &= \sigma(W_{xi}^T \hat{y}_t + W_{hi}^T h_{t-1} + b_i) \\
f_t &= \sigma(W_{xf}^T \hat{y}_t + W_{hf}^T h_{t-1} + b_f) \\
o_t &= \sigma(W_{xo}^T \hat{y}_t + W_{ho}^T h_{t-1} + b_o) \\
h_t &= o_t \otimes tanh(c_t)
\end{aligned} \tag{3.1}
$$

We then concatenated the predictions from the forward $\overrightarrow{\hat{y}}_t$ and backwards $\overleftarrow{\hat{y}}_t$ states (green block in Figure 3.5) and train a dense layer for classification, as we show on Equation 3.2, this is for predicting the same number of instances as in the groundtruth vector $\hat{y}$:

$$
\begin{aligned}
\overrightarrow{\hat{y}}_t &= \sigma\left(W_{\overrightarrow{x}\,\overrightarrow{o}}^T \hat{y}_t + W_{\overrightarrow{h}\,\overrightarrow{o}}^T \overrightarrow{h}_{t-1} + b_{\overrightarrow{o}}\right) \otimes tanh(\overrightarrow{c}_t) \\
\overleftarrow{\hat{y}}_t &= \sigma\left(W_{\overleftarrow{x}\,\overleftarrow{o}}^T \hat{y}_t + W_{\overleftarrow{h}\,\overleftarrow{o}}^T \overleftarrow{h}_{t-1} + b_{\overleftarrow{o}}\right) \otimes tanh(\overleftarrow{c}_t) \\
\hat{x}_t &= \left[\overrightarrow{\hat{y}}_t, \overleftarrow{\hat{y}}_t\right] \\
\overleftrightarrow{\hat{y}}_t &= W_y^T \hat{x}_t + b_y
\end{aligned} \tag{3.2}
$$

3.3.2 Loss Function for Visual and Temporal Recognition

For both networks, we utilize the standard softmax loss, which minimizes the error between the constructed probability distribution by the CNN and the groundtruth y, as denoted on Equations 3.3 and 3.4:

$$\mathcal{L}_{softmax-cnn} = -\sum y log(q(x)) \tag{3.3}$$

$$\mathcal{L}_{softmax-blstm} = -\sum y log(q(\hat{y})) \tag{3.4}$$

For the backbone CNN, the input $q(x)$ corresponds to the vector of probabilities, normalized by the softmax function q, and for the BLSTM, the input $\hat{y}$ corresponds to the normalized predictions from the dense layer in the backbone CNN.

3.4 Experimental Results

3.4.1 Datasets

The first dataset we used is T-LESS (Hodaň et al., 2017) (Figures 3.7(a) and 3.7(b)). It contains thirty industrial objects without any relevant texture or color, the objects present symmetries and mutual similarities between them and some are sub-parts of others. While T-LESS was collected in an *in-vitro* set-up by using a rotating table and controlled background and lighting conditions, it permit us to experiment with the viewpoints selection for training, since the full sphere of viewpoints is available. We use the total 37,578 training images and 69,546 images for testing. Furthermore, it allows to evaluate the generalization capability of a model trained with data containing nicely isolated objects with a black background and testing data with an increased complexity in clutter and occlusions.

The second dataset used is CORe50 (Lomonaco and Maltoni, 2017), which contains a set of fifty domestic objects belonging to ten categories: plug adapters, mobile phones, scissors, light bulbs, cans, glasses, balls, markers, cups and remote controls. The dataset is composed by a series of video clips, with 300 frames per video, showing an object being manipulated by a user, as we show in Figures 3.7(c) and 3.7(d). CORe50 allowed us to evaluate the performance of the model in the presence of occlusions produced by hands, unseen backgrounds and illumination conditions, which are well-suited for the *in-situ* set up that we are after. We used 119,894 images for training and 44,972 images for testing.

For evaluating each model, we use *a single video containing all the testing scene concatenated*, which means there are several objects across the video data, meaning that the models are required to adapt to different objects but also exploiting the temporal coherence correctly. As evaluation metric we use mean Average Precision (mAP) for T-LESS, consisting of averaging precision per instance and then globally. We choose this metric due to the data imbalance on the testing set. For instance, there are 8,000 images from object #1 while only 1000 images of object #30. We use Precision (P) for CORe50, since the testing set is balanced for all the instances. To compute the mAP, we first obtain the Precision by every instance, and then we average the results. For computing the Precision (P), we divide the number of correct predictions by the total number of testing samples.

(a) T-LESS Train set. Images were collected in a controlled environment.

(b) T-LESS Test set. Images contain a high level of occlusion and clutter.

(c) CORe50 Train set. Each training sequence depicts an object being manipulated by a user, showing an egocentric perspective to the object.

(d) CORe50 Test set. With the same dynamics as in the training videos, the testing set includes images collected in unseen environments with new illumination conditions and viewpoints.

Figure 3.7: T-LESS and CORe50 datasets images from (Hodaň et al., 2017) and (Lomonaco and Maltoni, 2017) respectively. The remaining objects are shown in Appendix A.

3.4.2 Baselines

Cumulative Moving Average: As a first baseline, we propose a simple sum-rule in which an average summation is performed on the *logits* vector $\hat{y}$ from the CNN, by a window size of n frames as denoted on Equation 3.5. We empirically found the value n with the best trade-off between producing coherent predictions and avoiding fusing predictions from different objects. To do so, we used the Feature Extraction model, and we report the precision using the finetuned model with the testing sequences concatenated, emulating a stream of images that an agent might receive during inference.

$$\hat{y}_t = \sum_{i=0}^{n} \frac{\hat{y}_{t-i}}{n} \tag{3.5}$$

As a side comment, we acknowledge that integrating spatial and temporal features require complex solutions, however, that does not mean that simple baselines, such as the cumulative average, should be ignored. On the contrary, simple but strong baselines as this one shows where the sophistication in the solutions should be focused on.

CortexNet: We utilize the original implementation from (Canziani, 2016), following the described training methodology. As an important note, CortexNet is a shallower model compared with Inception-ResNet-V2. Using CortexNet as baselines allow us to show the challenges of utilizing a custom CNN for learning generalizable features from few data.

We did not compare our model against Recurrent Ladder Networks since the authors do not provide code, nor there is a publicly available implementation of their work. Additionally, the model was designed to work with single-channel images of 32×32 pixels, while our datasets are full-scale RGB image size with a dimension of 299×299 pixels.

3.4.3 Implementation Details

To train our framework, we first train the backbone CNN. We utilize a pre-trained CNN with the ImageNet dataset, we fix all the weights from the Convolutional layers and train only the dense layer is trained. This process is commonly referred to *Feature Extractor* (Li and Hoiem, 2016) and it will be useful for choosing the hyperparameters for the temporal filters later on. Once the Feature Extraction model has converged, then a finetuning process takes place, where we re-train all the variables in the network until reaching convergence.

We use Inception-Resnet-V2 (Szegedy et al., 2016a) as the backbone CNN architecture for extracting and learning features since was the model with the highest recognition performance in ImageNet dataset and feasible to train with our GTX-1080 Ti GPUs with 11 GB memory. Smaller models such as MobileNets can be used for fitting more restricted hardware requirements. The model was initially trained on the Imagenet 2012 dataset, and for both re-training phases, we used softmax loss.

As hyperparameters for the Feature Extraction model, we use a batch size of 128 images and Root Mean Square Propagation algorithm for solving the optimization problem, as originally used in (Szegedy et al., 2016a), with the hyperparameters: weight decay $w_d = 0.0004$, learning rate from $l_r = 0.001$ to $l_r = 0.00001$, momentum $m = 0.9$ and $\epsilon = 1^{-10}$. For the *finetuning* phase, we selected the same optimizer but with smaller learning rates, starting at $l_r = 0.0001$ to $l_r = 0.000001$.

For training the BLSTM, we use the same training data used for the backbone CNN and similarly to (Canziani and Culurciello, 2017), all training video clips are concatenated and presented to the network until convergence. We used the Feature Extraction model since has a lower performance compared to the finetuned one, mainly because the majority of its weights come from a different dataset, and thus, it produces erroneous predictions on the training set, allowing the BLSTM to learn how to correct such incoherent predictions.

As hyperparameters for the BLSTM, we utilized the hyperbolic tan as non-linear activation, as it is commonly done for LSTMs (Graves et al., 2013). The weights and biases were initialized with a normal-random distribution and similarly with the finetuning model, we utilize root mean square propagation algorithm for solving the optimization problem, with the hyperparameters: weight decay $w_d = 0.0004$, learning rate from $l_r = 0.0001$ to $l_r = 0.000001$, decay $\rho = 0.9$, momentum $m = 0.9$ and $\epsilon = 1^{-10}$. Selecting such a small learning rate was crucial for training the BLSTM for avoiding the well-know *gradient vanishing problem* on Recurrent Neural Networks.

Our code was initially developed using TensorFlow 1.5 with the slim API, training scripts are available at (Lagunes-Fortiz, 2019a), including a version updated to TensorFlow 2.0, using the high-level API TF-Keras for using the pre-trained Inception-Resnet-V2 model and the Bidirectional LSTM implementation. All models were trained using a GTX-1080ti GPU and using Cuda 8.

Table 3.1: Precision achieved with different sampling methods and amount of training data (Without using temporal filters).

T-LESS	
Trajectory (10% training data)	mAP
Vertical	43.04
Circular	31.15
Sinusoidal	**43.26**
100% training data	46.82

CORe50	
Sampling Method	Precision
10% continuous frames	87.15
10% normal-random sampled	90.62
50% continuous frames	92.11
50% normal-random sampled	**94.16**
100% training data	94.32

3.4.4 Data Augmentation

In T-LESS, we deal with texture-minimal objects and differing training and testing distributions. We initially performed the recommended data augmentation procedure in (Donahue et al., 2014) regarding random cropping and modifications to color and contrast. Initial results showed, however, that the recommended augmentation for textured objects seems to produce inferior results when tested on texture-minimal objects. We thus do not use these data augmentation approaches for the remainder of the experiments in this and the remaining chapters. Applying random rotations *on-the-fly* resulted in a more useful on T-LESS to slightly boosting the performance. On CORe50, we did not use any data augmentation technique since the training data available shows the objects in different poses, lighting, and background conditions.

3.4.5 Recognition Performance by Sampling Method

In this experiment, we aim to characterize the effect of the variability in the training examples to build robust classifiers, by varying the amount of training samples and viewpoints. T-LESS allowed us to test how training with views from different sampling methods affect the recognition performance. The images were obtained following the sampling methods previously described. The size of the training data from all trajectories is 10% relative to the initial full-sphere set. For CORe50, we tested frame-to-frame sampling versus normal-sampling from each training video clip, with a total of images of 10% and 50% relative to the total amount of training data available. We run each training session three times for 150 epochs.

Table 3.1 contains the results of using trajectories for collecting data versus a full-POV sphere. For these experiments, we utilize the CNN model only, without utilizing any of the temporal filters from Figure 3.5. We run every training session three times, showing only the best run. As expected, results show that using more data leads to the

best performance, however models trained with data containing high variability, such images coming from the vertical or sinusoidal, offered close performance to a model trained with ten times more data and allows faster convergence times for the training the CNN.

Additionally, Table 3.1 contains the results on CORe50 when different amounts of data are used, comparing frame-to-frame versus normal-random sampling from the video clips. Similarly, the model trained with more data leads to the best performance and normal-sampled slightly outperforms frame-to-frame sampling. However, interestingly the difference between 100 and 10 percent of training is only 3.7%. These results show that diversity of the training data is critical for training CNNs efficiently.

3.4.6 Temporal Integration

In Table 3.2, we present the mean average precision when the temporal filters are added. We present the results when the models were trained with 100% and 10% of the available training data. For T-LESS, the 10% of data available comes from the sinusoidal sampling method since offered the best performance, and for CORe50, we utilize the 10% from the random-normal sampling method since outperformed the frame-by-frame one.

For T-LESS, related to the use of Cumulative Moving Average, we show the best performance achieved by a window with a size of 25. For the BLSTM, we show the best performance achieved by a length of 500 cells with 300 neurons. For CORe50, we report the best performance achieved by a CMA with a window size of 40 and a Bidirectional LSTM with length 300 and 200 neurons. Experiments for finding these values are presented in the next section.

As a first result, by adding the cumulative moving average (CMA) and the Bidirectional LSTM (BLSTM), the models trained with a 10% of the total available training data achieved a similar level of performance as the one trained with the 100%. On the other hand, the performance achieved by CortexNet is considerably lower (around 20% less, on average) than the Inception-ResNet-v2 backbone, this is in part due to the shallower architecture of CortexNet, but also due to the fact than the model only considers the current and previous frame to produce the predictions.

Surprisingly, the CMA performs not far from the sophisticated BLSTM (with less than 2% inferior recognition performance), however, on a closer inspection such as the one presented in Figures 3.8 and 3.9, this is due to the good performance from the backbone CNN for correctly predicting the object in each scenes in the majority of the testing sequence and producing misclassification under specif situations such

Table 3.2: Precision achieved by CNNs and using Temporal Filters.

	T-LESS		CORe50	
Model	100% training data (mAP)	10 % training data (mAP) (sinusoidal traj.)	100% training data (P)	10% training data (random sampling) (P)
CNN (w/o temporal filter)	46.82	43.26	94.32	90.62
CortexNet (Canziani and Culurciello, 2017)	33.25	28.11	74.41	61.35
CNN + CMA (ours)	52.46	47.94	97.63	94.43
CNN + BLSTM (ours)	**56.23**	**55.04**	**99.13**	**97.92**

as ambiguous viewpoints and partial occlusion, specially for the T-LESS dataset, in which the initial frames of each testing sequence corresponds to isolated top views with occlusions happening on the latest frames.

If we revert the order in the testing sequence, and by doing this, presenting an occluded start followed by an isolated ending as in the first part in Figure 3.8, then the BLSTM shows a better performance by producing correct predictions after the frames with occlusion, while the CMA required more frames to recover due to the moving average. We also highlight that the performance of the BLSTM on T-LESS is not very robust against strong occlusions most likely because during the training we used the isolated images without invariances, while for CORe50 the training and testing sets depicts similar conditions than the BLSTM can exploit, in the sense that shows changes in the background and occlusions, but depicts the same type of occlusions and manipulations.

In Figures 3.8 and 3.9, we present frame-to-frame predictions produced by each model used (CNN, CortexNet, CNN + Cumulative Moving Average, and CNN + BLSTM). In each Figure, we show images with the groundtruth and a given testing frame showing the object pretended to be recognized. Below, we present the predictions from every model, to maintain the plots readable, we only show the Top-10 objects detected in that given sequence.

In Figure 3.9, we show how both filters resulted useful for correcting the erroneous predictions from the CNN, caused by ambiguity at the given object's pose. This is, some views are quite similar among the five mugs (objects label 41-45), such as the top view. The ambiguity was alleviated by using information from previous frames with more discriminative information. The first 300 frames corresponded to the black scissors (object 13) and represented by the blue line on the predictions plot. The rest 300 frames corresponds to the red and green mug (object 41), represented by a red line.

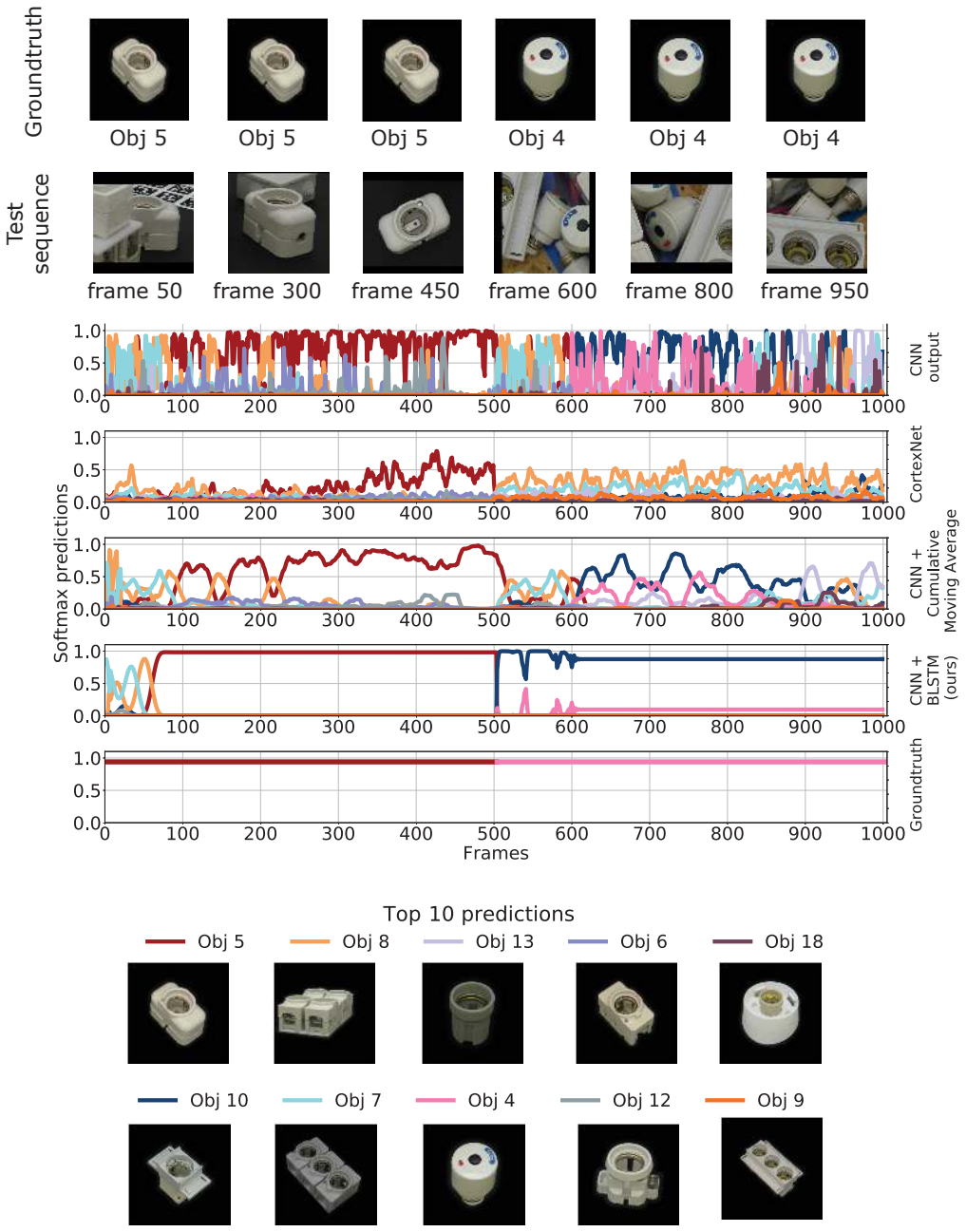


Figure 3.8: Fragment of T-LESS testing sequence showing objects 4 and 5. Highly occluded conditions resulted quite challenging for the CNN and this could not be alleviated by the use of Temporal Filters. Please use the electronic version for a closer view and refer to Appendix A for numbering of the instances.

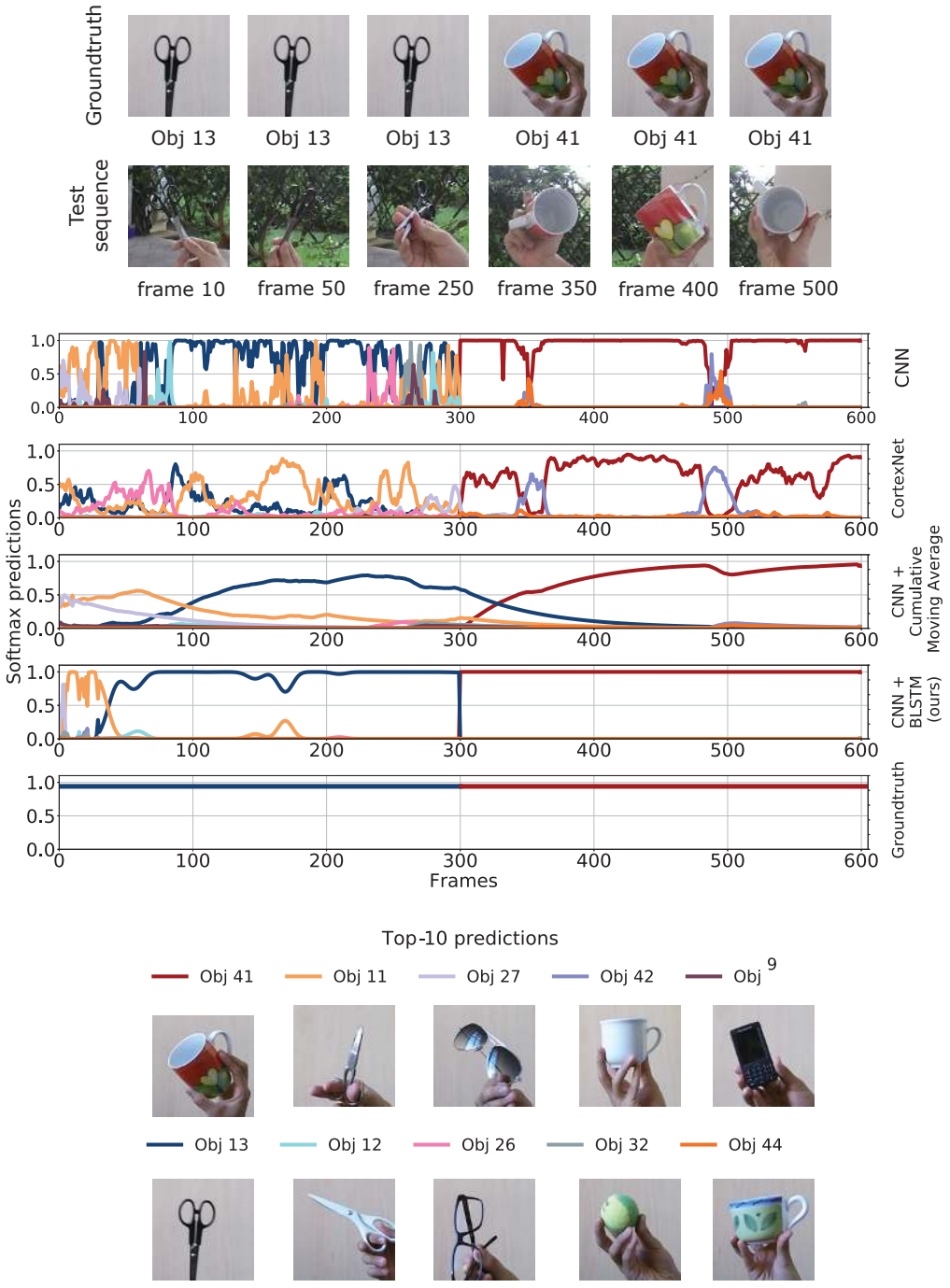


Figure 3.9: Fragment of Core50 testing sequence, showing object 13 -black scissors- and 41 -red and green mug-. The temporal filters resulted useful for correcting misclassifications caused by object's pose ambiguity. Please use the electronic version for a closer view and refer to Appendix A for the numbering of the instances.

In Figure 3.8, we show how the BLSTM performs slightly better than the CMA, in this case, by recovering faster after a set of erroneous predictions from the CNN, due to clutter and occluded conditions. Additionally, we present how neither of the filters can compensate from a majority of incorrect predictions caused in this case by strong occlusion from other objects. This limitation from the CNN of dealing with clutter and occlusions is more acute for minimal-texture objects, especially when objects are parts of other objects. We attribute this big gap in performance between the dataset to these CNN limitations and how to overcome them, remains as an open research question.

In Figure 3.10, we present the normalized confusion matrices between the CNN and the BLSTM, using the 10% of the data from each dataset. The numbering used on the matrices is consistent with the instances ID number presented in Appendix A.

T-LESS (Figures 3.10(a) and 3.10(b)), resulted in a more challenging task for the CNN than the objects in CORe50, we believe this is explained in part by the texture-minimal characteristic on the objects which makes them easier to be confused on cluttered, partial and fully occluded conditions, *e.g.*, objects like the texture-less box #27 are easily misclassified when stronger features from other objects such as the holes from object #9 appears on the image. Additionally, training and testing present contrasting conditions, while the training data does not contain any occlusion or different background conditions, the testing images present strong occlusions and clutter, making generalization more challenging to achieve.

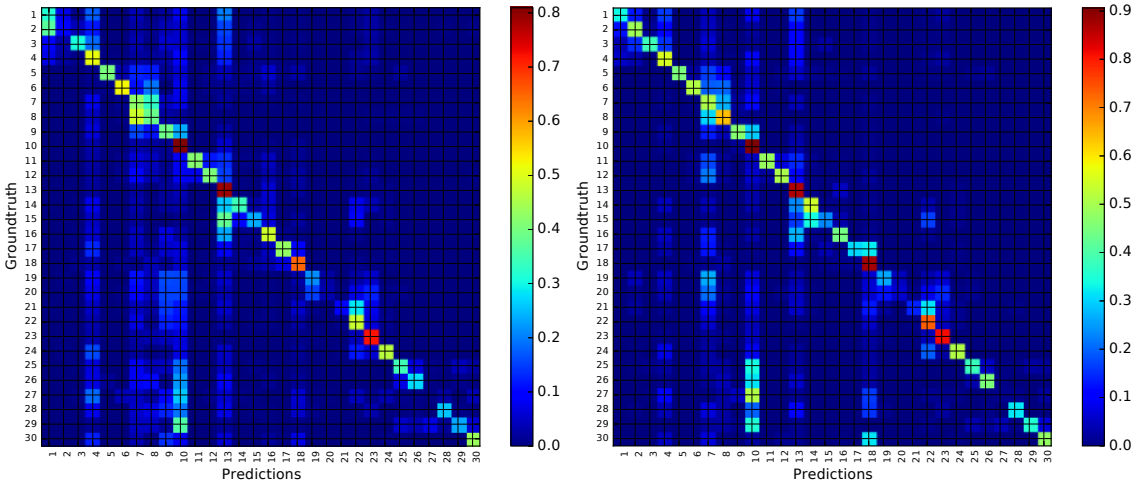
On CORe50 (Figures 3.10(c) and 3.10(d)), we notice that misclassification occurs mostly between objects from the same classes. For example, for the case of glasses (objects labeled from 26-30) present the same geometry and visual information when they are shown from the top view, then only the temples and top-bar are visible, and thus, the model can get easily confused. This is indeed a challenging recognition situation that was partially solved with the temporal filters.

3.4.7 Ablation Studies

Bidirectional LSTM Design

We varied the number of cells, corresponding to the number of frames seen by the BLSTM, from 100 to 500 cells (Figure 3.12(a)) and we varied the number of neurons on the gates, going from 100 to 500 (Figure 3.12(c)).

For T-LESS, the BLSTM with the best performance was the one with a length of 500 cells with 300 neurons. For CORe50 the highest precision was the configuration with a



(a) CNN trained with T-LESS

(b) Model with CNN + BLSTM with T-LESS

(c) CNN trained with CORe50

(d) Model with CNN + BLSTM with CORe50

Figure 3.10: Normalized confusion matrices with frame-by-frame evaluation using 100% of training data, the instances ID numbers are consistent with Appendix A.

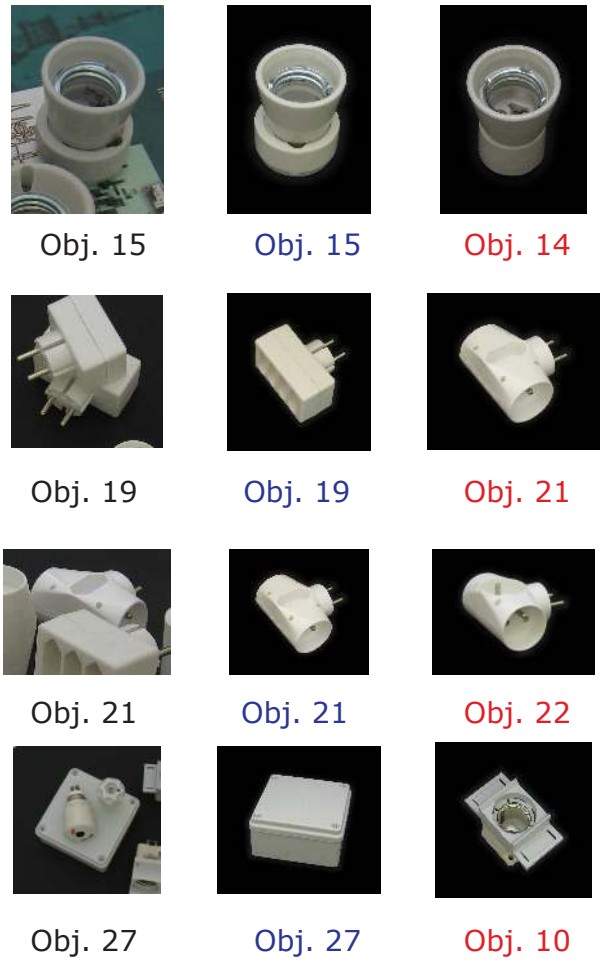
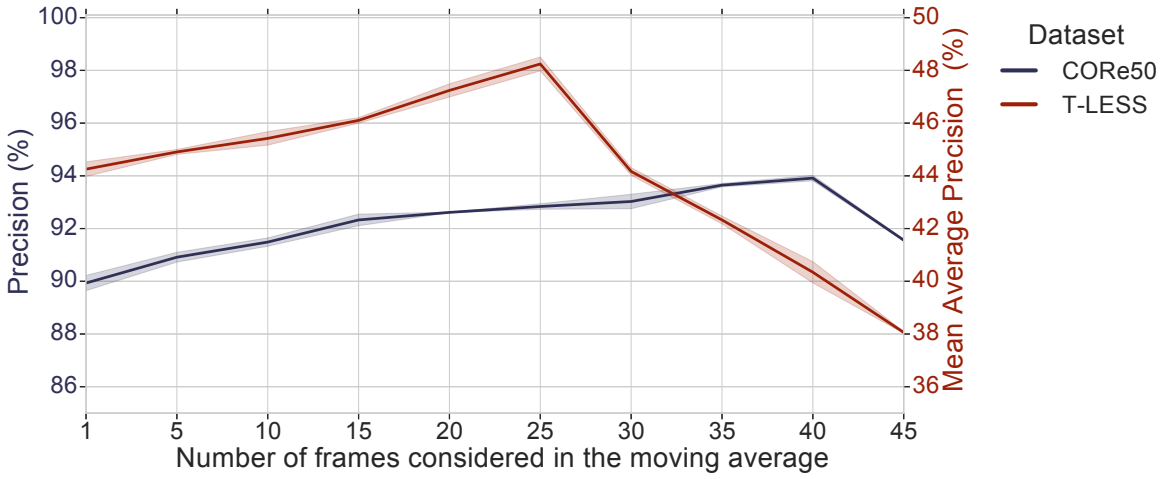
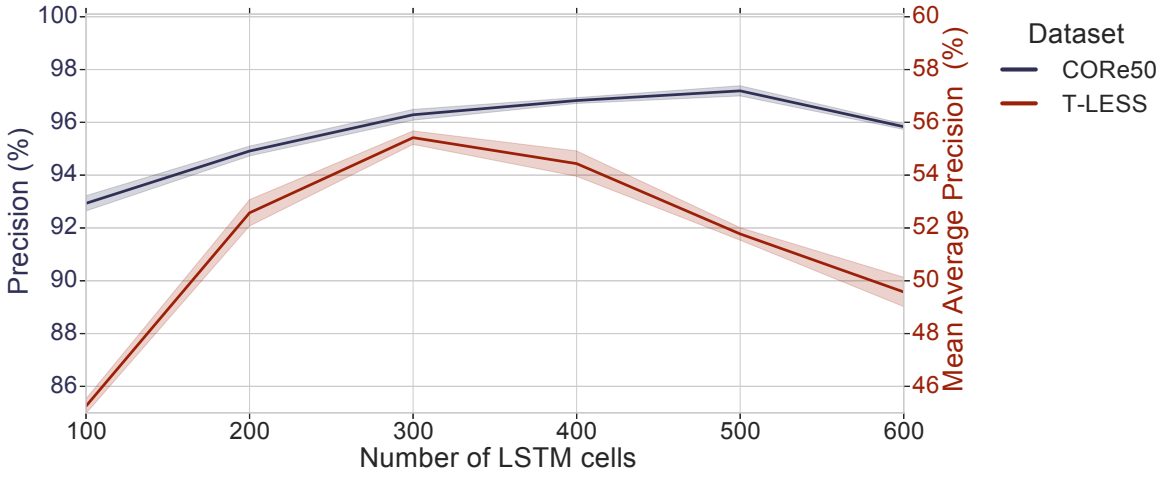


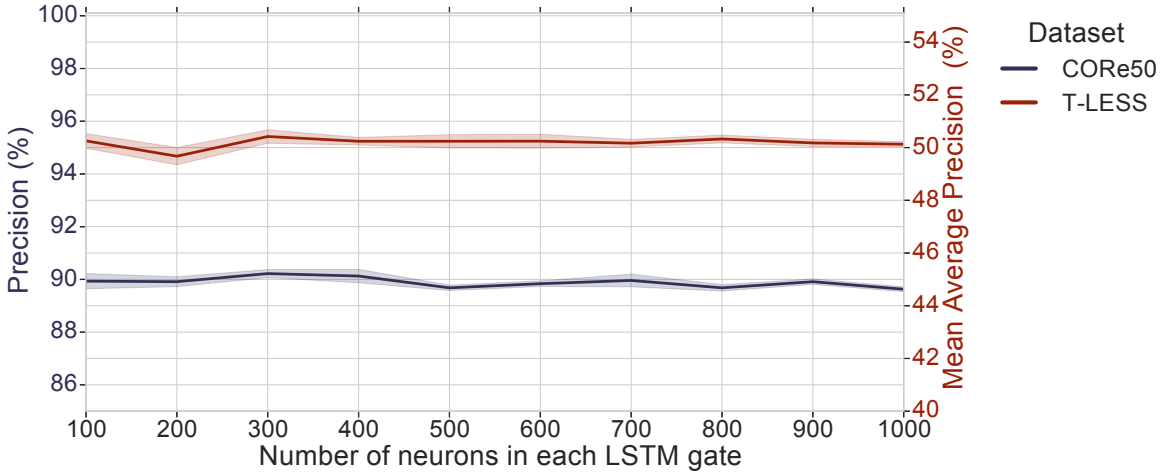
Figure 3.11: Examples of misclassification in the T-LESS dataset, caused by confusion between similar instances and the clutter. In each row, we present a test frame, the groundtruth with blue, and the prediction of the model with red.



(a) Selecting of the hyperparameter n in the cumulative moving average



(b) Selecting the number of cells LSTM cells in the recurrent network.



(c) Selecting the number of neurons in each cell block, in the recurrent network.

Figure 3.12: Ablations studies to select the best configuration for the BLSTM and the cumulative moving average.

length of 300 cells and 200 neurons. These results indicate that the BLSTM found that objects are alternating every 500 frames in the testing sequence in T-LESS and every 300 frames in CORe50.

Moving Averange Tunning

For selecting the size of the averaging window with the best precision, we varied the windows size from 1 frame to 45 frames in steps of 5 frames, as shown in Figure 3.12(a). For the T-LESS dataset, the CMA with the highest performance was the one achieved by a size of 25 frames. For CORe50, using 40 frames led to the best performance, which the latter being consistent with the one reported on (Lomonaco and Maltoni, 2017).

Both temporal filters boosted the performance on the Precision (mAP on T-LESS and AP on CORe50); related to the BLSTM, the results are consistent with works like (Zamir et al., 2017) and (Clark et al., 2017), in which the performance gets better by using more LSTM cells and with the number of neurons being not as relevant as the number of cells, for both T-LESS and CORe50 selecting the longest length available lead to the best results, while choosing a *small* number of neurons of around 200 worked as good a one trained with 600.

3.5 Summary

In this chapter, we propose an approach to multi-view instance-level object recognition that combines convolutional and recurrent neural networks for exploiting the *temporal coherence* present in video data. We first evaluate sampling methodologies for using training data efficiently, concluding that the variability in the training examples, this is, presenting examples to the network that depicts the objects under different conditions, allows the model to learn generalizable features efficiently, as opposed to using hundreds of examples depicting the object under similar conditions. We then present two temporal filters for exploiting the assumption that on video data the same object remains present over a number of consecutive frames, consisting of a cumulative moving average and a machine learning approach using recurrent neural networks. We evaluate these temporal filters on T-LESS and CORe50. We show how this way of learning could be especially useful when few data points are desired for training the models, accelerating the training process, which is useful for agents exploring the world in front of them and when they need to react and use these objects without delay, and achieves higher recognition performance than state-of-the-art approaches such as CortexNet. We conclude that the

BLSTM resulted slightly more useful for exploiting the temporal coherence (around 2%higher precision), but it does require to be trained, while the simpler CMA filter shows itself useful, with the main disadvantage being the dimensionality of the window fusion. Overall our methods show an improvement on the precision of *state-of-the-art* CNNs that do not exploit the temporal element by its own, with a much higher performance (over 20% higher precision) than the shallower CortexNet architecture. With our temporal filters, we achieve a precision of 99.13% for CORe50 and a mean average precision of 56.23% in T-LESS. The gap in performance is partially explained by the differing training and testing conditions in T-LESS, and we address this limitation of CNNs in the next chapter.

CHAPTER 4

LEARNING EMBEDDINGS FOR DOMAIN GENERALIZATION

In the previous chapter, the use of temporal filters on top of CNNs allowed near perfect recognition for the CORe50 dataset (99.13%). However, even with temporal filtering, our model achieved only 56.23% precision in T-LESS, primarily due to the differing training and testing conditions on the dataset. Therefore, in this chapter we focus on learning generizable features from a few data belonging to a single source domain, such as synthetic images, or data collected in a specif location, and that can be used when the domain has changed, and now depicts new imaging conditions, clutter, differing scales, etc. To do so, we propose the used of a multi-task CNN that performs image classification and image generation to achieved domain generalization.

4.1 Introduction

In Chapter 3, we showed how the use of temporal filters on top of *state-of-the-art* CNNs help increasing the robustness against the commonly encountered changes in perspective, scale, illumination, object's pose and noise. However, the temporal filters were only useful when the CNN was trained with several examples from the same instance, as in the CORe50 dataset with training images obtained from eight different environments. In contrast, in the *in-situ learning* set up that we are after, we aim to use data collected from a single training scene. Therefore, in this chapter we propose a model that learns

generalizable representations, using data depicting objects from a single environment and that can be used to recognize such objects under unseen conditions.

An example between differing training and testing environments is depicted in Figure 4.1, where the model has to learn from clean synthetic data, and it is deployed on real images with clutter and occlusions. Additional examples of differing training and testing environments are: data collected indoors versus outdoors, data collected from high-resolution vs. low resolution sensors and data collected in a controlled environment vs data from the real-world. This problem is referred as *domain generalization* (Li et al., 2018a) and we consider the case of training the model with a ***single*** domain, motivated by the expensive procedure of collecting data from multiple domains and the need of learning from few available data in the context of robotics and autonomous systems applications.

With this goal in mind, we propose a Supervised Residual Autoencoder (S-RAE) for learning features that are discriminative between instances but generalizable across domains. To do so, the model utilizes a multi-task reconstruction and classification loss. We perform inference by utilizing k-nearest neighbors on the embeddings space, in order to evaluate how discriminative the embeddings are, that is, data points from the same instance should be close to each other and further away otherwise.

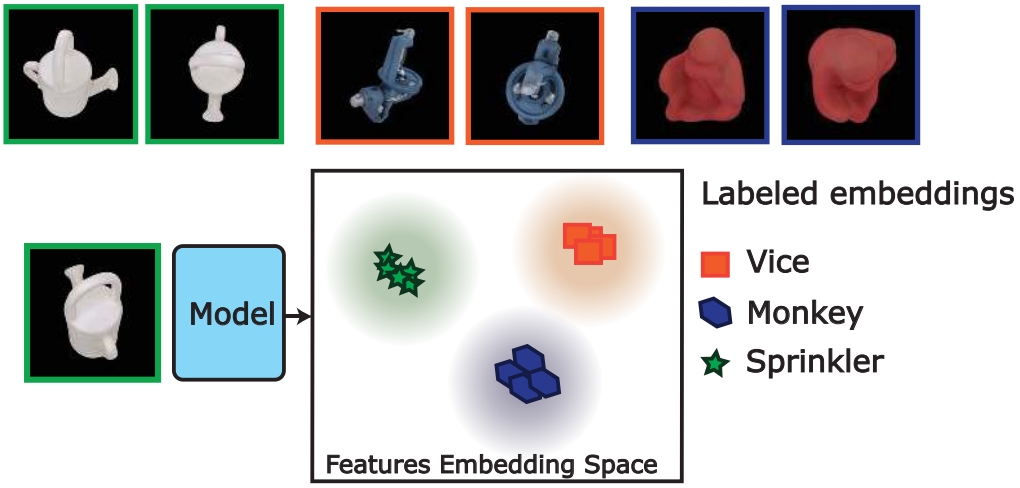
We evaluate our approach on datasets tailored for instance-level object recognition with differing training and testing conditions: Linemode, Tejani, T-LESS, Toyota Light, TU Dresden Light, ToyBox, Rutgers-APC, and CORe50.

4.2 Related Work

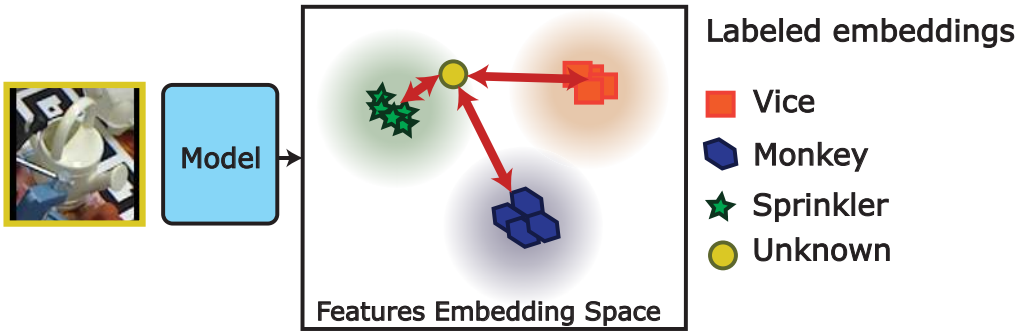
The problem of the differing train and test data distributions has been approached in two variants: Domain adaptation and domain generalization (Ghifary, 2016).

For domain adaptation, it assumes access to data points from both source and target domains (Liu et al., 2018b). In contrast, in domain generalization there is no availability of target domains, but multiple source domains might be available (Li et al., 2018a). Within the latter approach, we consider the case of learning from a single source domain, motivated by the expensive procedure of collecting data from multiple domains in robotics and autonomous systems applications, where systems need to react based on few observations of objects on varied and novel contexts.

Here we concentrate our review on *domain generalization* approaches for object recognition, including metric learning and multi-task learning models, which integrate image



(a) Training



(b) Inference

Figure 4.1: Problem description. The model is required to learn to recognize objects from synthetic data with a clean background (a) and has to generate discriminative features under unseen conditions, such as clutter and occlusions (b).

generation and classification for learning disentangled and transferable representations as in our work.

A first approach is Metric Learning, where a model is trained to learn the concept of *similarity* by minimizing the distance between embeddings from the same instance and separating embeddings from different ones by a margin. Top performing approaches to metric learning employ *state-of-the-art* CNNs (Song et al., 2017; Bellet et al., 2013). Within the metric learning approaches, triplet loss (Schroff et al., 2015) posits as the most widely used approach for metric learning (Zhe et al., 2018), and consist of a model with three CNN branches that encourages a distance constraint between similar and dissimilar images simultaneously.

More recently, center loss (Wen et al., 2016) proposes the same goal of minimizing

the intraclass distance of the embeddings by learning a center for each instance and penalizing the distances between the embeddings and their corresponding instance centers. In order to ovoid cumbersome sampling techniques, the centering loss is combined with a supervised signal (cross-entropy loss) as a regularizer. Following the idea of making embeddings more discriminative, (He et al., 2018) combines the triplet and center loss for multi-view object retrieval, while also keeping the supervised loss as a regularizer. The idea of clustering data points and using a supervised loss as regularizer has also been applied successfully for few-shot learning as in Prototypical Networks (Snell et al., 2017), indicating it is indeed a powerful technique for generating discriminative features.

Another successful strategy for generating discriminative embeddings is by incorporating additional tasks to the model. The underlying assumption for most multi-task learning algorithms is that different tasks are related to each other and the related tasks act as regularizers to the main one (Yin and Liu, 2018). There is a vast literature on multi-task learning, therefore we only present approaches with image classification as the primary task and image generation as a secondary task. While all the following approaches report an increase in accuracy by adding the additional tasks, the caveat is how to combine (*weight*) such tasks during the optimization process, since an unweighted combination of the losses often leads to a decrease in the performance due to the high penalization from the regularizer.

A first example is (Zhi et al., 2017), which combines object classification and orientation estimation for 3D Object Detection. In (Yin and Liu, 2018) and (Wang et al., 2017d) the main task is person identification and pose, illumination and expression as secondary tasks. In these approaches, a hyperparameter λ is found empirically for weighting each loss term.

DR-GAN (Tran et al., 2017) performs both classification and image generation, being an equivalent of our model, but using an adversarial network for image generation as oppose to an autoencoder. It follows the concept of building a supervised and multi-task Discriminator and, similarly to us, adding image generation as a secondary task. DR-GAN utilizes an adversarial network for generating images, as opposed to a convolutional autoencoder with mean squared error loss as in our work. Additionally, the authors do not make any assumption about the number of domains used during training, in contrast with another *state-of-the-art* adversarial approaches (Chopra et al., 2013).

A contemporary work to this thesis introduces Supervised Autoencoders (SAE) (Le et al., 2018), where the authors provide a theoretical demonstration on how reconstructing images with a conventional autoencoder aids to achieve higher generalization for

classification tasks. Although they utilize mean squared error for classification, and not cross-entropy as in this work, we find similar results on using the image generation as a regularizer. That is, adding the reconstruction loss does not harm the recognition performance and can significantly improve generalization. While in this work we address the usefulness of supervised autoencoders for learning generalizable representations from a single domain, experiments conducted by (Le et al., 2018), show their usefulness for learning from multiple domains.

In contrast to (Le et al., 2018), our model uses a deeper architecture with full-scale real images, it utilizes the Nearest Neighbors search in the embeddings space for inference, and utilizes a single residual connection between encoder and decoder.

We use image reconstruction by a conventional convolutional autoencoder, which was initially proposed for image compression and de-noising (Du et al., 2017), since we do not focus on achieving a high-quality reconstructed. In this regard, several regularizers have been proposed in order to avoid over-fitting, with weigh decay and the Jacobian matrix (Rifai et al., 2011) being popular choices.

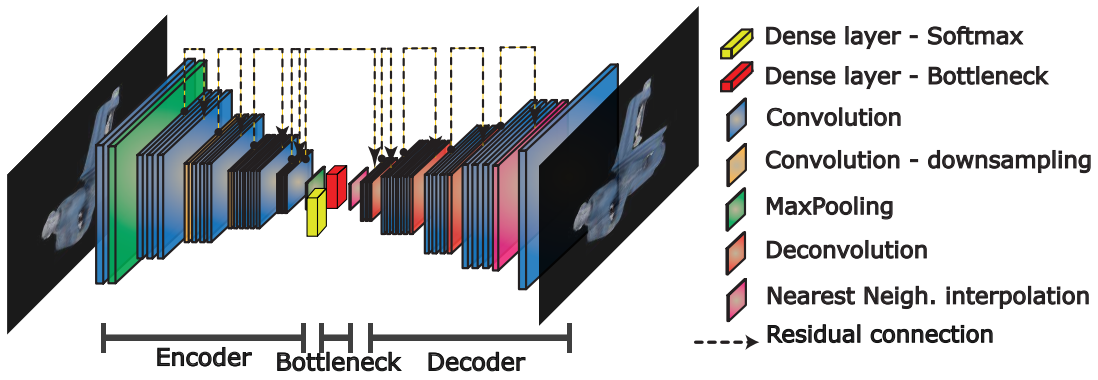
4.3 Proposed Method

We propose a Supervised Residual Autoencoder (S-RAE) for learning generalizable embeddings through the process of jointly learning to classify and reconstruct images. We now explain the main components in our model and how we use it to perform object recognition.

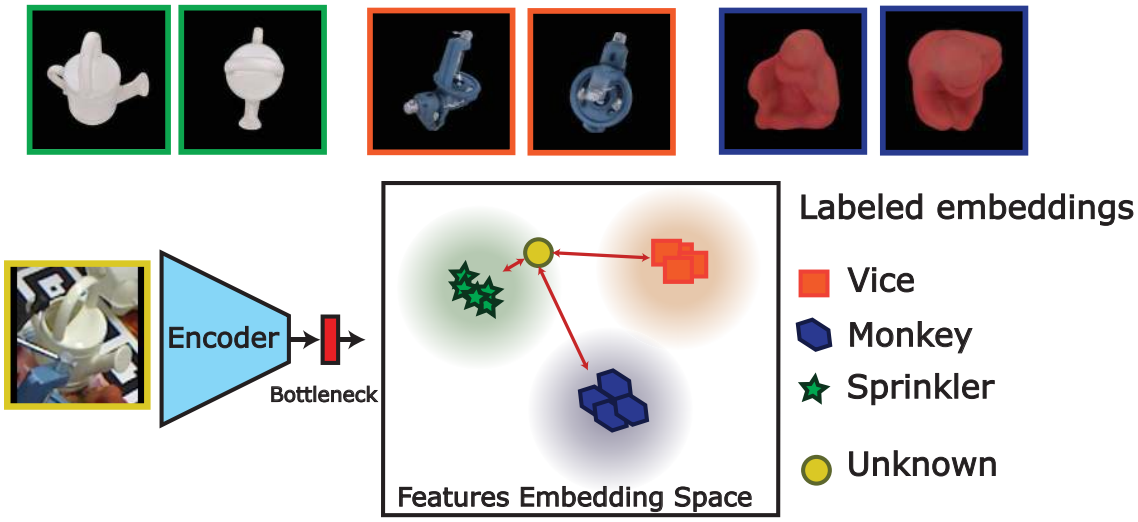
4.3.1 Architecture Design

Our architecture is inspired by *state-of-the-art* convolutional autoencoders like U-Net (Ronneberger et al., 2015), SegNet (Vijay et al., 2017), and PointGrid (Le and Duan, 2018), in which the encoder corresponds to a *state-of-the-art* CNN model (VGG (Simonyan and Zisserman, 2015) for the aforementioned works) and the decoder mirrors the Encoder configuration, replacing downsampling operations for upsampling ones and skipping connections across encoder and decoder .

In our model (Fig 4.2(a)), the encoder corresponds to ResNet-50 (He et al., 2016). We flatten the last feature map from the encoder and connect it to a fully connected layer. This dense layer is the bottleneck (*code*) in our deep convolutional autoencoder, and it is shared by a fully supervised classification layer and the input to the Decoder.



(a) Training



(b) Inference

Figure 4.2: S-RAE: Supervised Residual Autoencoder for learning *generalizable* embeddings. a) Architecture design. b) Deployment: We perform inference by a k-nearest neighbors search over labeled embeddings, obtained from training images, and a query embedding with an unknown label.

In the Decoder, the bottleneck is connected to a fully connected layer for bringing the dimensionality back to the one in the last convolutional block size (2048 elements for ResNet 50). We then mirrored the convolution layers from the encoder and utilize transpose convolutions (*deconvolutions*) with 1×1 kernel for upsampling the feature maps between the residual blocks, where convolutions with stride two were used for downsampling and nearest neighbors interpolation for upsampling the feature maps that were obtained by max-pool or average pooling operations. After the first upsampling operation, the one increasing the feature maps from $1 \times 1 \times 2048$ to $7 \times 7 \times 2048$ (height $\times$ width $\times$ channels), we added a skipping connection from the last convolutional feature map in the encoder and first convolutional block in the decoder to ease the optimization by avoiding having a feature map with a high content of zeros or repeated values. We show in the ablation studies section that adding more skipping connections across the encoder and decoder, leads to a decrease in performance due to the weaker learned representation in the bottleneck.

4.3.2 A Loss Function for Learning Generalizable Embeddings

The loss function in our Supervised Residual Autoencoder (S-RAE) is defined by:

$$\mathscr{L}_{S\text{-}RAE} = \mathscr{L}_{reconstruction} + \mathscr{L}_{softmax} \tag{4.1}$$

$$\mathscr{L}_{reconstruction} = \mathscr{L}_{MSE} = \frac{1}{m}\sum_{i=i}^{m}(X_i - X_{i_{rec}})^2 \tag{4.2}$$

$$\mathscr{L}_{softmax} = -\sum y log(q(x)) \tag{4.3}$$

Where the reconstruction loss in the image space is the mean squared error between an input image X_i, with m pixels, and its reconstruction by the decoder $X_{i_{rec}}$ (Equation 4.2). The softmax loss (Equation 4.3) utilizes the groundtruth labels y and features x from the classification layer (yellow block in Figure 4.2(a)), and acts as a regulizer by imposing a separability constraint into the embeddings. Our motivation behind this, is to ensure that the manifolds found by the autoencoder are separable (which will be reflected in the k-nearest neighbors search). In (Le et al., 2018), the authors utilize sums of squared errors, as opposed to cross-entropy, to derivate their theoretical demonstration on the

usefulness of using the reconstruction loss as a regularizer. We show in the ablation studies that selecting cross-entropy loss leads to higher accuracy. Noticeably, there is no weighting term for the reconstructions loss, as early experiments show no improvements over the unweighted loss term.

4.3.3 Inference by Matching Embedding

During deployment, we use the encoder and embeddings $\hat{x}$ from the bottleneck (red dense layer in Figure 4.2(b)). Using the embeddings from the bottleneck is a critical aspect of our work since the bottleneck was never trained with the distances between embeddings, but by using the combination of the reconstruction and classification losses. We project the training images into the embedding space and perform inference by matching embeddings with known labels and a query embedding from a testing image. That is formally expressed in Equation 4.4, where $\hat{x}_i$ represents an embedding generated by the bottleneck, K represents the number of neighbors to check and $N_K(\hat{x}_i)$ the index of the K neighbors, I is an indicator function expressed in Equation 4.5.

$$P(y = c|\hat{x}_i, K) = \frac{1}{K} \sum_{i \in N_K(\hat{x}_i)} I(y_i = c) \tag{4.4}$$

$$I(e) = \begin{cases} 1 \text{ if } e \text{ is True} \\ 0 \text{ if } e \text{ is False} \end{cases} \tag{4.5}$$

Performing inference by matching embeddings allows us to evaluate how discriminant the embeddings from each model are. Inference by the softmax and bottleneck layers is also possible, as we show in the experiments section. We selected the standard version of k-nearest neighbors (with $k = 5$) and not the weighted version, to produce comparable results with (Zeng et al., 2018).

As shown in Figure 4.2(b)), we remove the decoder since in this chapter we are interested in the recognition performance only, however, the Decoder has the potential to be used for image segmentation as in SegNet (Vijay et al., 2017) and U-Net (Ronneberger et al., 2015).

4.4 Experimental Results

In order to test the usefulness of our approach for generalizing under unseen domains, we tested our method across datasets where the training examples are obtained from a single domain and the testing examples depicts a different one. We then describe the datasets, baselines and ablation studies that we conducted.

4.4.1 Datasets

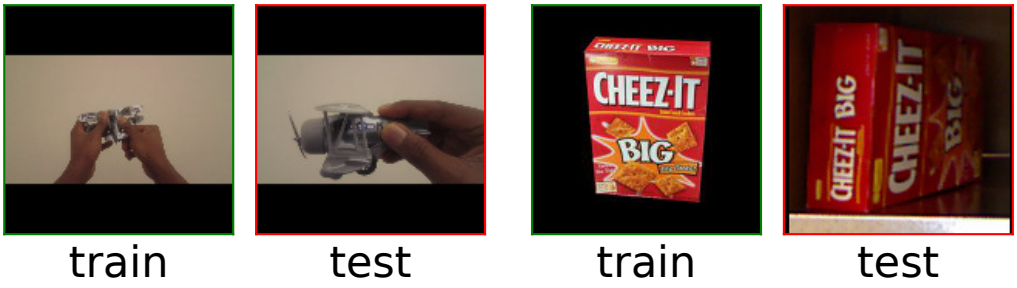
We used a total of eight datasets, six of them (Linemod, Tejani, RU-APC, T-LESS, TUD-L, and TOY-L) were initially proposed for 6D pose estimation and offer the challenge of learning from few data with clean background and controlled illumination and generalize in environments depicting high clutter and occlusions, which is the interest of this work. For these datasets, we used the unified format proposed by (Hodan et al., 2018), which was designed for an easy benchmarking of these datasets. This benchmark consist of 15,563 images for training and 7,450 images for testing.

We also utilize ToyBox (Wang et al., 2017c) which contains 360 objects and allows us to evaluate how well the model scales for learning a higher number of instances. We selected the hodgepodge sessions for training and the x,y,z transformations for testing, which includes changes in scale and partial occlusions. We sub-sampled the original dataset set by selecting a 10% (with a normal random distribution) of the original dataset, in order to fit it in our memory requirements. We denote ToyBox* as the sub-sampled version that we used in this work. This resulted in using 2,466 images for training and 6,977 images for testing.

The remaining dataset is CORe50 (Lomonaco and Maltoni, 2017), which was proposed for continuous/lifelong learning. It shows 50 objects across eleven environments, which allows us to test the generalization capabilities not only to unseen objects' conditions but also new environments. We chose the first scenario for training and the conventional scenarios #3, #7 and #10 for testing. This results in 14,989 images for training and 44,972 images for testing.

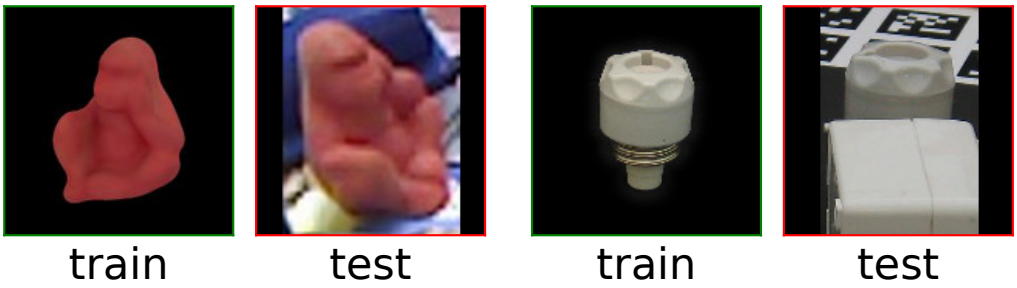
In Figure 4.3, we show an example between training and testing domains. In Table 4.1 we describe of the differences between training and testing environments.

(a) TU-Dresden (b) Tejani



(c) ToyBox (d) Rutgers-APC

(e) CORe50 (f) Toyota-light



(g) Linemod (h) T-LESS

Figure 4.3: Contrast between train and testing conditions for each dataset.

Table 4.1: Differences between train and test environments.

Datasets	Objects	Training Domain	Testing Domain
(a) TU Dresden (Hodan et al., 2018)	3	real images were taken with a specific background and illumination	clutter and differing lighting conditions
(b) Tejani (Tejani et al., 2014)	6	synthetic images, clean background	clutter and slight occlusions
(c) Toybox* (Wang et al., 2017c)	360	real images, clean background	changes in scale and mild occlusions
(d) Rutgers - APC (Rennie et al., 2016)	14	synthetic images, clean background	clutter and slight occlusions
(e) CORe50 (Lomonaco and Maltoni, 2017)	50	real images were taken with a specific background and illumination	clutter and differing object poses and lighting conditions
(f) Toyota - Light (Hodan et al., 2018)	21	synthetic images, clean background	different lighting conditions
(g) Linemode (Hinterstoisser et al., 2013)	15	synthetic images, clean background	heavily cluttered and mild occlusions
(h) T-LESS (Hodaň et al., 2017)	30	real images, clean background	heavily cluttered and heavy occlusions

4.4.2 Baselines

All trained models, but DR-GAN, contain the same number of parameters during deployment, consisting of ResNet-50 with an additional fully connected layer of 128 elements. In DR-GAN, we did not add an additional layer in order to keep the model as close as possible to the author's implementation. We did not compare against the supervised autoencoders proposed by (Le et al., 2018) since the model was designed for small-scale images (32x32 images) and has much less capacity, with only two convolutional layers compared to 50 layers in ResNet-50; however, an equivalent model with higher capacity is presented in the ablation studies. Each baseline was implemented as follows:

ImageNet Features: As a first baseline, we explore the usefulness of utilizing the features generated by a pre-trained ResNet-50 with the ImageNet dataset. To perform the k-nearest neighbors search, we utilize the flattened features from the last convolution, consisting of 2048 elements. Since ResNet50 does not have intermediate fully connected layers, we follow the methodology proposed by (Zeng et al., 2018) which consists of taking the feature map after the average pooling layer and flatten it to a 1-D vector with 2048 elements (this dimension resulted from using a resolution of 224×224 pixels). This baseline allowed us to explore if a model trained with a large dataset can produce discriminative embeddings for multi-view object recognition.

Triplet Loss: The triplet loss is a widely effective metric for discriminative embeddings (Zhe et al., 2018). For training the model, we select a margin value of $m = 4$ as performed by (He et al., 2018), and we performed hard mining in the form of selecting the 20% most difficult examples per mini-batch and using them in the subsequent batches.

Center Loss: This supervised loss was designed for generating discriminative embeddings by clustering data points to a leaned center point from each class. We selected the hyperparameters $\lambda = 0.1$ and $\alpha = 0.005$ in the loss function (Wen et al., 2016). We utilized the PyTorch implementation from (Zhou, 2018).

DR-GAN: We replaced the original encoder, based on CASIA-Net (Yi et al., 2014), for

ResNet-50 so the encoder's capacity can be comparable across all models. We removed poses prediction, since some of our datasets do not provide such information, and keep the label prediction in the discriminator and image reconstruction in the generator. We utilize the PyTorch implementation from (Junh, 2019).

4.4.3 Implementation Details

We initialize the transposed convolution filters using the Kaiming-normal method (He et al., 2015), the fully connected layer with a uniform random distribution. Convolutions in the Encoder and Decoder are initialized from a pre-trained Resnet-50 pretrained with Imagenet. Thus, our model works with RGB images with size 224×224. All convolution layers are followed by batch normalization and ReLU layers with a skipping connection as ResNet-50 (He et al., 2016). The transposed convolutions are followed a batch normalization and linear activations.

We choose a dimensionality of 128 elements for the bottleneck, in order to reduce the redundant information from the last convolutional feature map. As studied in (Vassileios Balntas and Mikolajczyk, 2016) and (Taigman et al., 2014), a dimensionality of 128 can offer a comparable recognition performance as a 4096 one, indicating that the latter might containing highly redundant information.

For training our model, we used mini-batches of 55 images, as it that was the largest batch size possible in our GTX 1080Ti GPUs with 11GB, we used stochastic gradient descent with a learning rate of 1×10^{-3} and momentum 0.9. We developed our models using PyTorch 1.0 (Paszke et al., 2017), trained on GTX 1080Ti GPUs, and utilized the multicore k-nearest neighbors algorithm from the sci-kit learn library (Pedregosa et al., 2011). Code with training scripts is available at (Lagunes-Fortiz, 2019a).

4.4.4 Object Recognition Accuracy

In Table 4.2 we present a comparison of each model across the eight datasets. Features were extracted from the bottleneck, as shown in Figure 4.2(b). We utilize k-nearest neighbors with $k = 5$ and trained every model three times using the same train/testing split. We show the mean and standard error accuracy, in order to evaluate the stability of each model.

Our S-RAE achieved an average 5.23% higher performance with respect to the center loss, the closest baseline. It resulted particularly useful for the Linemode dataset with a 12.12% increase with respect to the ImageNet features, the closest baseline. For Tejani

and T-LESS, it achieved a competitive performance by around 1% behind the closest baselines, being center loss and ImageNet Features respectively.

In general, all the models struggled for T-LESS and Linemode datasets, indicating that models trained to recognize texture-less objects from clean data and testing on highly clutter and occluded environments, it is still a big limitation for current domain generalization approaches. Possible solutions to mitigate this limited performance is by including additional information such as depth and/or bounding box/masks estimation, which can be used to enforce attention to the object's geometry and reduce the attention to the background's appearance.

The model trained with center loss offered a competitive performance, with an average 5.82% behind our model; however, from Table 4.2, it is noticeable that it struggled for generalizing its centering effect over datasets with a higher number of classes. Our method has an additional advantage over the center loss approach, by not requiring searching the hyperparameters that balance softmax and centering loss respectably, since our losses can be combined linearly, nor it is necessary to compute the embeddings distances during training.

In DR-GAN, the embeddings generated by the encoder, which is part of the generator, were not as discriminative as our approach, with an average 22% lower performance. The lack of discriminability on the encoder's embedding is likely to the fact that the classification is done by the discriminator and the image generation by the generator, and each component is trained by a separated optimizer and not as a multi-task model where a single optimizer is used and the second task is considered as a regularizer. Thus, the discriminator acts as a dense layer trained purely with the cross-entropy loss as in in conventional CNNs.

From Figure 4.5(a), we show how the embeddings generated by our method lead to a higher recognition accuracy, compared with the embeddings generated by only utilizing the cross-entropy or triplet loss. Overall, we can infer that our model is successful in finding a low-dimensionality latent representation (manifold) for every object, which is theoretically invariant to small perturbations.

In Table 4.3 we present a comparison of each model across the eight datasets with the use of the cumulative moving average filter, like the one proposed in Chapter 3, with a window size of 20 frames. Since the CNN is not predicting labels as in Chapter 3, we utilize the predictions from the nearest neighbors classifier. Similarly to the previous experiments, the features were extracted from the bottleneck, as shown in Figure 4.2(b), we used the models from the previous table and utilizing k-nearest neighbors with $k = 5$.

Table 4.2: Average accuracy and standard error.

Datasets	ImageNet Features	Triplet Network	Center Loss	DR-GAN	**S-RAE (ours)**
(a) TU Dresden Light	89.95	94.21 ± 0.58	99.60 ± 0.11	94.40 ± 1.5	**99.82 ± 0.06**
(b) Tejani	66.49	46.79 ± 4.11	**95.89 ± 2.92**	65.28 ± 1.61	94.75 ± 0.34
(c) ToyBox*	52.46	65.02 ± 1.49	75.49 ± 1.20	59.43 ± 0.71	**80.25 ± 0.67**
(d) Rutgers-APC	34.96	33.95 ± 2.71	67.72 ± 2.34	43.11 ± 1.95	**74.95 ± 0.77**
(e) Core50	29.91	41.22 ± 1.26	65.81 ± 3.78	53.51 ± 1.82	**72.17 ± 1.52**
(f) Toyota Light	50.23	15.21 ± 1.41	55.87 ± 2.98	47.62 ± 1.53	**61.13 ± 0.39**
(g) Linemod	38.37	39.02 ± 5.08	34.76 ± 4.28	22.10 ± 2.64	**51.35 ± 2.12**
(h) T-LESS	**42.97**	27.21 ± 1.08	39.23 ± 4.75	19.49 ± 0.88	41.79 ± 0.33
Average	50.67	45.33 ± 2.10	66.79 ± 2.80	50.61 ± 1.58	**72.02 ± 1.12**

Table 4.3: % Accuracy with temporal smoothing using a cumulative moving average.

Datasets	ImageNet Features	Triplet Network	Center Loss	DR-GAN	**S-RAE (ours)**
(a) TU Dresden Light	95.21	98.61 ± 0.68	99.80 ± 0.21	98.60 ± 0.21	**99.92 ± 0.04**
(b) Tejani	70.33	47.82 ± 3.87	**97.89 ± 2.12**	68.38 ± 1.01	96.81 ± 0.12
(c) ToyBox*	70.25	75.24 ± 1.38	84.12 ± 1.09	65.33 ± 0.64	**89.16 ± 0.56**
(d) Rutgers-APC	37.89	36.45 ± 2.56	70.24 ± 1.24	46.51 ± 1.24	**81.63 ± 0.47**
(e) Core50	33.43	51.53 ± 1.16	69.52 ± 2.58	57.91 ± 1.62	**76.97 ± 1.12**
(f) Toyota Light	56.43	16.51 ± 1.31	59.20 ± 2.31	53.12 ± 1.31	**69.03 ± 0.32**
(g) Linemod	47.12	42.02 ± 4.88	36.26 ± 3.58	24.34 ± 2.21	**56.15 ± 2.08**
(h) T-LESS	**54.03**	27.81 ± 1.02	43.10 ± 3.21	23.10 ± 0.21	46.39 ± 0.33
Average	58.08	49.49 ± 2.11	70.02 ± 2.04	54.78 ± 1.05	**77.01 ± 0.63**

For this experiment we did not aim for finding an optimal value for the Cumulative Moving Average, instead, we wanted to explore how utilizing a generic temporal filter impacts the recognition performance across all datasets.

By using the temporal filter, the recognition performance increased by an average 5% across all the models. Indicating that temporal filters are also useful on CNNs trained with limited data, in the form of data collected from a single domain, and trained for manifold learning, in contrast with label prediction as in Chapter 3.

As discussed in Chapter 3, a disadvantage of the moving average is finding the optimal value for the number of frames to be considered in the summation, which we found by a greedy search in each dataset. In Chapter 6, we propose an alternative method that integrates the idea of temporal modeling with manifold learning, in order to learn discriminative features that are aware of the temporal coherence present in nearby frames.

Results from Tables 4.2 and 4.3 indicates that our proposed Residual Supervised Autoencoder achieves *state-of-the-art* performance for learning generalizable features

from training data collected in a single domain, which is a core aspect in the *in-situ learning* that we are after. The other aspect to consider in this dissertation, is the efficiency for learning new instances, that is, learning new instances without finetuning the model. The motivation being the desirability of a model capable of learning new objects within the natural place where the objects are utilized, meaning that high-end computing resources might not be available for finetune the model.

To test such capability in our model, we utilize the methodology proposed by Zeng et al. (2018), which aims to measure the discriminability of the features generated from seen and unseen objects. To do so, each dataset is divided into a *novel* and *known* sets, where two-thirds of the classes composed the *known* set are used for training the model and the remaining third correspond to the *novel* set and it is used for recognizing new instances without finetuning the model, and performing the nearest neighbors search in the embeddings space. The known and novel subsets are different from the typical training/validation/test splits, since in the novel set, contain instances that were not seen during training.

The methodology consists of utilizing the model trained with the known set and evaluate how well the model generalizes for producing discriminative features of objects not seen during training. For associating the embeddings between training and testing images the nearest neighbors search, similar to the one performed in the previous experiments, is performed in the embeddings space. The instances for novel and known sets were randomly chosen and are stated in the Appendix A.

Noticeably, finetuning the last layer for producing labels from the objects in the novel split, is still not a viable option for in-situ learning, since the full original training data is required in order to avoid forgetting the previous learned instances. Furthermore, constantly finetuning a model with an ever-growing dataset is an inefficient solution, both in energy and time required, that motivate us for a model that does not require access to the original training set or perform any sort of finetuning at all.

In Table 4.4, we show the performance achieved by every model when two thirds of the dataset are used for training and the remaining third for learning objects *on-the-fly*. We show the recognition performance when a query image is known to belong to a known object (rows denoted as *Known*), when it belongs to a novel object (rows denoted as *Novel*) and the general case where there is no prior information about the query object and features from both known and novel object are project in the embeddings space for performing the nearest neighbors search (rows denoted as *Known & Novel*).

The **Triplet Network** was the model with worst performance overall. Still, its

Table 4.4: Learning novel objects *on-the-fly* % Accuracy top-1 recognition

Objects	ImageNet Features	Triplet Network	Center Loss	DR-GAN	S-RAE (ours)
(a) TU Dresden Light					
Known	99.22	99.34	**99.86**	99.63	99.70
Novel	100	100	100	100	100
Known & Novel	**99.00**	89.95	98.19	95.01	96.66
(b) Tejani					
Known	66.60	64.90	73.79	65.51	**81.44**
Novel	**98.23**	71.43	69.19	93.49	92.02
Known & Novel	**66.49**	49.39	57.70	52.5	59.13
(c) ToyBox*					
Known	64.44	43.74	**80.51**	77.34	**79.04**
Novel	55.18	53.22	**79.01**	77.25	78.87
Known & Novel	52.46	43.74	69.36	72.34	**74.21**
(d) Rutgers-APC					
Known	37.70	43.59	56.74	**65.86**	63.19
Novel	52.23	42.60	62.26	71.77	**73.47**
Known & Novel	34.96	35.81	42.84	38.41	**44.65**
(e) Core50					
Known	59.95	33.69	91.59	**93.32**	92.99
Novel	66.82	41.28	94.22	95.35	**96.11**
Known & Novel	54.12	29.91	77.25	**86.55**	86.39
(f) Toyota Light					
Known	56.50	35.96	**54.80**	**61.61**	56.81
Novel	**69.55**	43.89	44.24	63.89	54.53
Known & Novel	**50.23**	12.92	32.44	32.20	38.21
(g) Linemod					
Known	42.68	52.60	**60.54**	49.46	56.01
Novel	**58.45**	53.40	35.37	53.68	44.71
Known & Novel	**38.37**	31.60	36.77	30.47	34.70
(h) T-LESS					
Known	**48.95**	15.05	44.85	36.18	35.32
Novel	**51.95**	18.64	33.13	35.03	37.51
Known & Novel	**42.97**	12.97	28.04	28.27	27.55
Average					
Known	59.05	48.61	70.34	68.61	**70.56**
Novel	69.05	53.05	64.67	**73.80**	72.15
Known & Novel	54.83	38.28	55.32	54.48	**57.58**

performance is much higher than a random chance, and for the case of learning few objects it offered competitive results, considering that is being trained by using semi-supervised examples.

Using the **ImageNet Features** from the last convolution layer from a pre-trained model with the ImageNet dataset, lead to better results for learning texture-less objects with clean background, but cluttered and occluded testing scenarios. This was not the case for learning objects in the texture and instance level recognition, since the pre-trained model was trained for learning in the *class* level and not instances. Additionally, the embeddings generated by the pre-trained model are 16x bigger (2048 vs 128) which led to a slower inference time.

The **DR-GAN** approach generalized well when there are few objects projected in the metric space, since the embeddings tend to be more spread due to the lack of additional regulizers constraining the learned features. As a result of this, it has overall lower performance when known and novel embeddings are projected in the same space.

The model trained with **center loss** offered a competitive performance for known objects, however, from Table 4.4, it is noticeable that it struggled for generalizing its *centering* effect over unseen examples and thus, the combined known and novel objects embeddings lead to a lower performance.

Our **Residual Supervised Autoencoder** had the best performance with the biggest datasets (Toybox and CORe50), had a similar performance for the known objects as center loss, with the advantage of not searching the hyperparameters alpha and lambda that balances softmax and centering loss respectably, since our losses can be combined directly. However, it also struggled for generalizing in the texture-less datasets, as it performed worse than utilizing the ImageNet features.

For the TLESS dataset, we attempted to alleviate the over-fitting when learning from few and texture-less images, with two approaches: (a) freezing parts of the models and trained only the last and deconvolution layers and (b) performing regularization in the form of weight decay ($wd = 1 \times 10^{-3}$ to $wd = 1 \times 10^{-8}$) and dropout ($p = 0.1$ to $p = 0.7$). However, neither of these approaches led to higher accuracies, indicating that a more complex solution is necessary for this particular dataset.

Noticeably, all the model struggled in the case of the feature space with embeddings from known and novel objects. As an example of this, our model, achieves an average precision of 70.56% and 72.15% for known and novel splits respectively. However, when the two sets are projected into the embeddings space the performance decreases to 57.58%. Similarly, the model trained with center loss achieves an average precision

of 70.34% and 64.67% for known and novel splits respectively and a lower 55.32% for combined known and novel splits. Thus, in Chapter 5 we propose a model that deals better with the more general case of having embeddings from known and novel objects in the same metric space.

In order to learn more about the limitations of our model for learning novel objects *on-the-fly* we show images generated from both a known and unseen object in Figure 4.4. We utilized a the Tejani dataset for showing a case where our model achieves a higher and worse performance than the baselines. Figures 4.4(a) and 4.4(b) show an object seen during training, where the model is able to reconstruct the input image similar and during deployment ignores the clutter in the background and reconstruct only the information regarding the object of interest. On the other hand, Figures 4.4(c) and 4.4(d) show an object never seen during training. In this case the model is not able to fully reconstruct the input image, omitting parts of the object from both training and testing images, which ultimately was reflected in lower recognition performance. This indicates that our S-RAE does not have a good generalization for unseen objects and finetuning is the recommended strategy for learning novel objects.

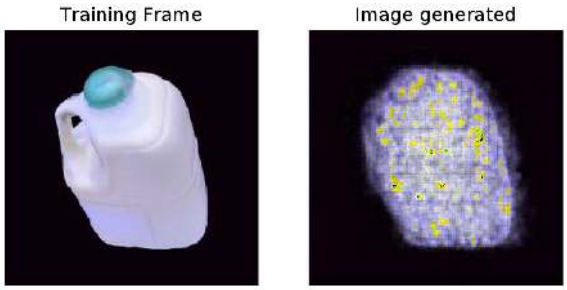
4.4.5 Ablation Studies

We performed the following modifications in our model, that led to the configuration with the best overall performance. Our ablations studies cover the use of different upsampling operations such as transpose convolutions and nearest neighbors interpolations, and the use of alternative loss functions, including the classification and reconstruction losses separately:

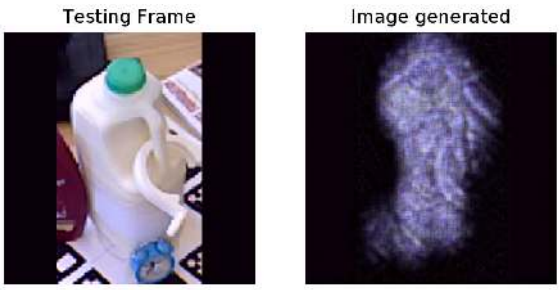
Architechture Design: In Figure 4.5(a), we present several configurations of our model. First, we remove the classification loss, resulting in a conventional autoencoder, and reconstruction loss, resulting in the original ResNet-50 model for classification.

We then explore the use of upsampling operations, considering a decoder with transpose convolutions as the only upsampling operation, and using interpolation operations only. We found a combination of both led to an average 10% increase for our datasets.

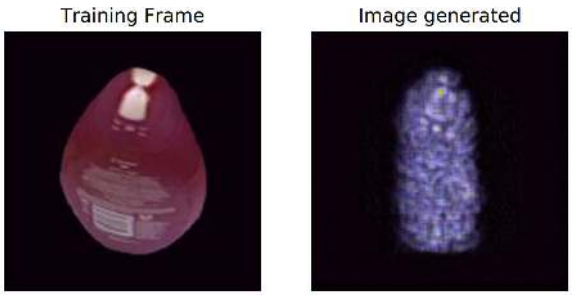
We also present the effect of the skipping connection between the last convolutional block and the first upsampling operation, using this configuration resulted in an average 1.23% increased than not using skipping connection at all and 12% increase compared to using skipping connections across all ResNet blocks.



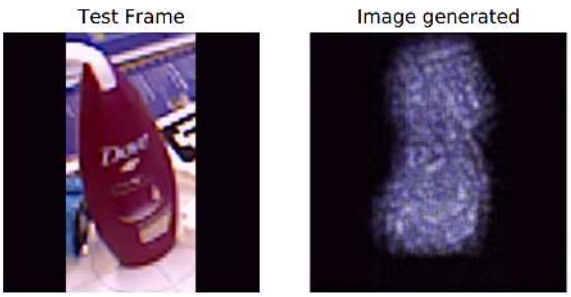
(a) Object seen during training. The model is able to reconstruct the shape of the input image.



(b) Boost in performance: The reconstructed testing image shows that the image was encoded using the object's appearance only and ignoring the background.



(c) Object seen during training. The model is able to reconstruct the shape of the input image.



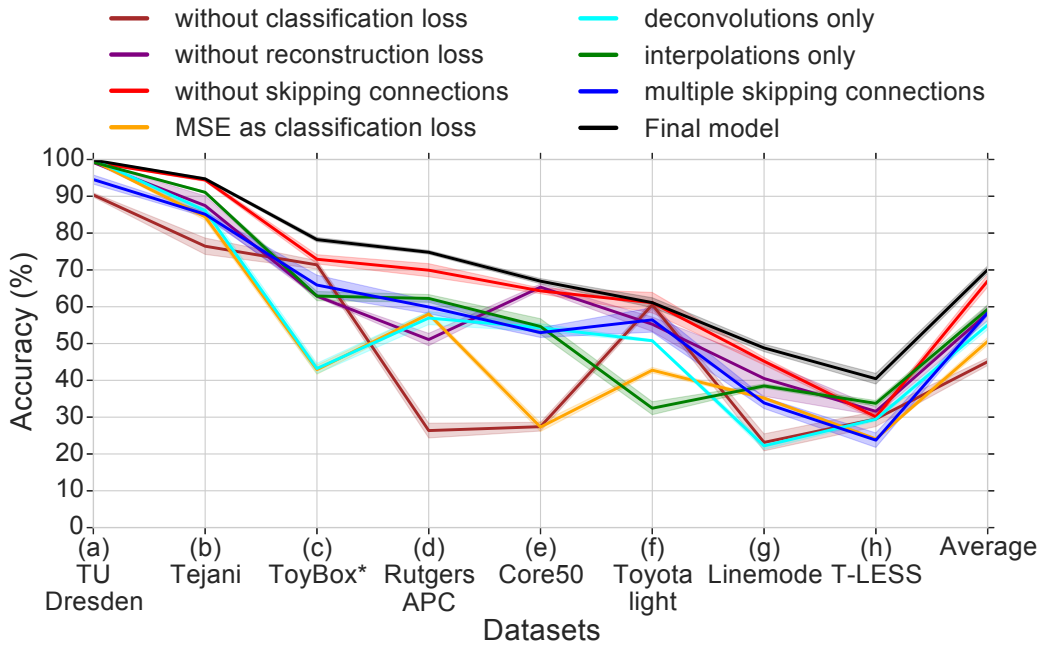
(d) Boost in performance: The reconstructed testing image shows that the image was encoded using the object's appearance only and ignoring the background.

Figure 4.4: Advantages and limitations of our residual supervised autoencoder for learning novel objects *on-the-fly*.

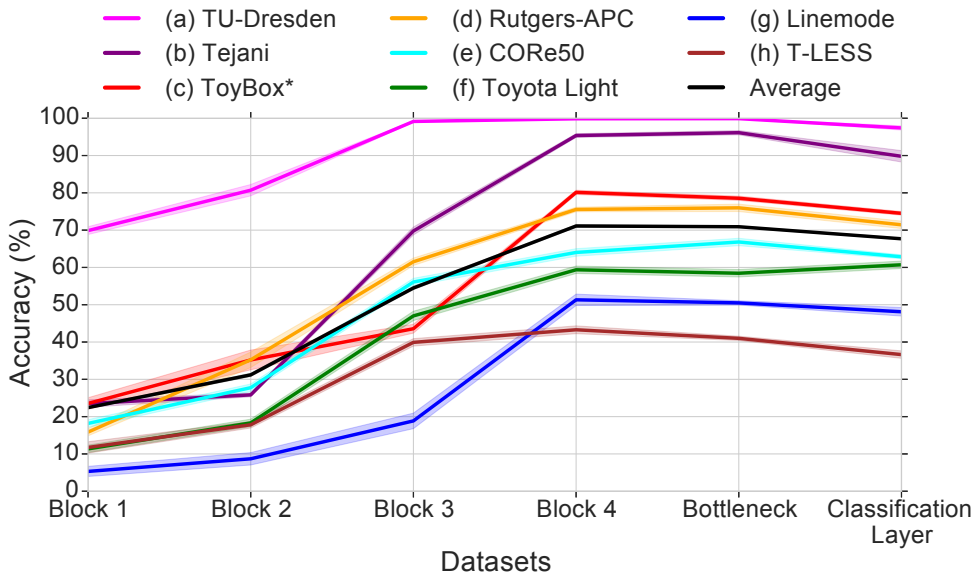
We include the effect of utilizing the sum of square errors for classification, as proposed in the supervised autoencoders by (Le et al., 2018), which led to a 20% drop in performance compared with the cross-entropy loss.

The final version of our model, which includes a mixture of interpolations and transposed convolutions, a single skipping connection, and trained with a combination of cross-entropy and mean squared error, led to higher performance across all datasets but Toyota-light. In this case, the model achieves a more stable performance but with a slight decrease of 2.8% compared with one trained model without a skipping connection.

Feature Extraction and embeddings matching: Additionally, We explore if it is possible to achieve better classification performance from earlier layers of the model. We perform classification by utilizing the features from each ResNet block and the classification layer used for regularized the model. For the ResNet blocks we utilize



(a) Architecture design.



(b) Feature Extraction.

Figure 4.5: Ablation studies include modifications in the architecture (a) and performance achieved by each ResNet block in the encoder and the classification layer (b).

k-nearest neighbors with $k = 5$ for matching embeddings from training images. For the classification layer we softmax the predictions and chose the prediction with the highest score. However, a disadvantage of utilizing the k-nearest neighbors search is that the inference time keeps increasing, as more training samples are acquired.

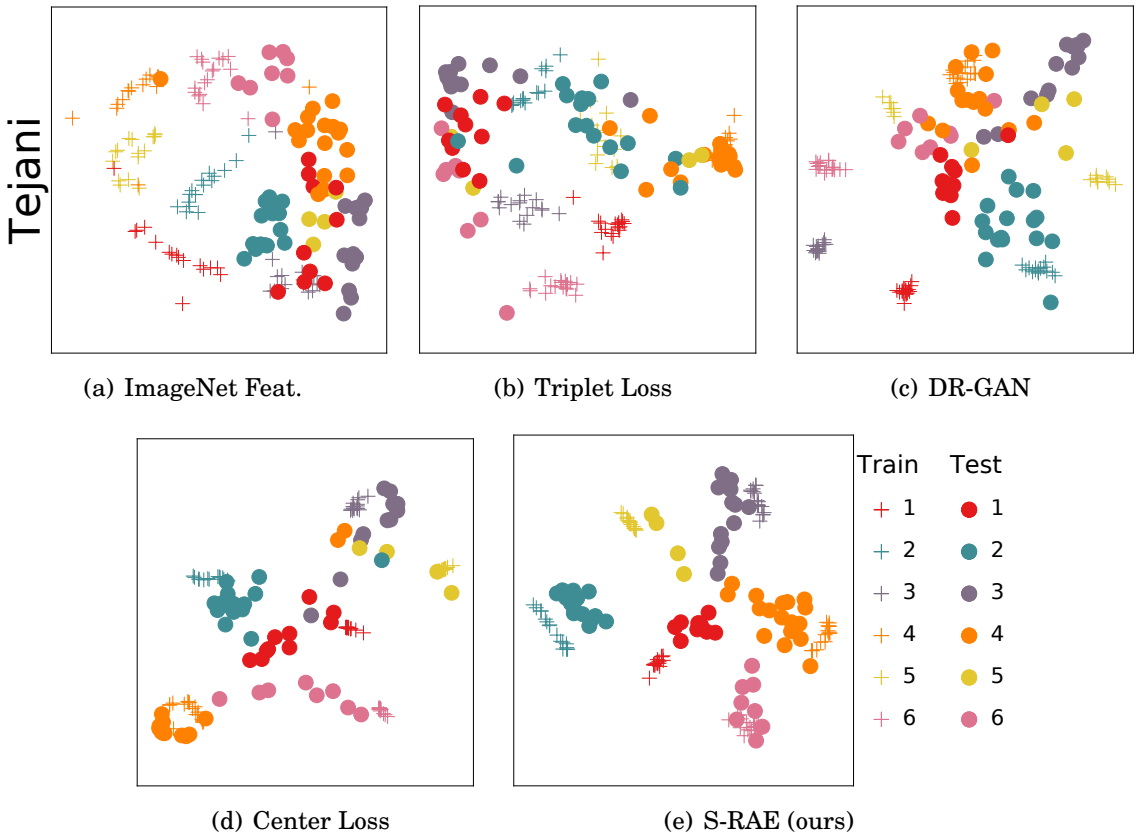
Every experiment was run three times. From Figure 4.5(b) we can see how extracting features from the bottleneck resulted into an overall slightly increase of 1.3% compared with the classification layer and comparable performance, with an average 0.43% decrease from the 16x larger Block 4. Probabilities from the classification layer were obtained by performing a softmax operation in the predicted labels.

To further verify our intuition of a successful generation of discriminative features, we present t-SNE visualizations for Tejani (Figure 4.6), Rutgers-APC (Figure 4.7) and for CORe50 dataset (Figure 4.8). The objects from the legends are shown in the Appendix A.

In these Figures, we can appreciate how the triplet loss, and DR-GAN, succeed in generating discriminative embeddings in the training examples but struggled to bring together embeddings from the same class but different domain. Center loss, and our proposed S-RAE, were able to bring together embeddings from different domains. Noticeably, embeddings from novel objects tend to be more spread, compared to the embeddings from known objects. This spreading has a negative effect when embeddings from known and novel object are present in the feature space and the proposed method in Chapter 5 ameliorates the feature's spreading in novel object.

4.5 Summary

In this chapter, we address the problem of learning to recognize objects using a few example data and generalize to unseen environments, which is a core aspect in the *in-situ learning* scenario that we are after. The features learned by our model are discriminative between classes and aim to be indistinguishable between domains. Our approach consists of a Supervised Residual Autoencoder (S-RAE) that combines image classification and reconstruction. We empirically evaluate our approach on eight real-world object recognition datasets used for Robotics and Computer Vision recognition tasks and achieving an average 5.23% higher performance compared with the closest baseline and over 20% higher performance compared with an *state-of-the-art* DR-GAN, an adversarial network approach and center loss, a supervised metric learning approach for learning discriminative features. Our proposed S-RAE however, assumes the availability of computational resources for learning new instances, which can be a big limitation for

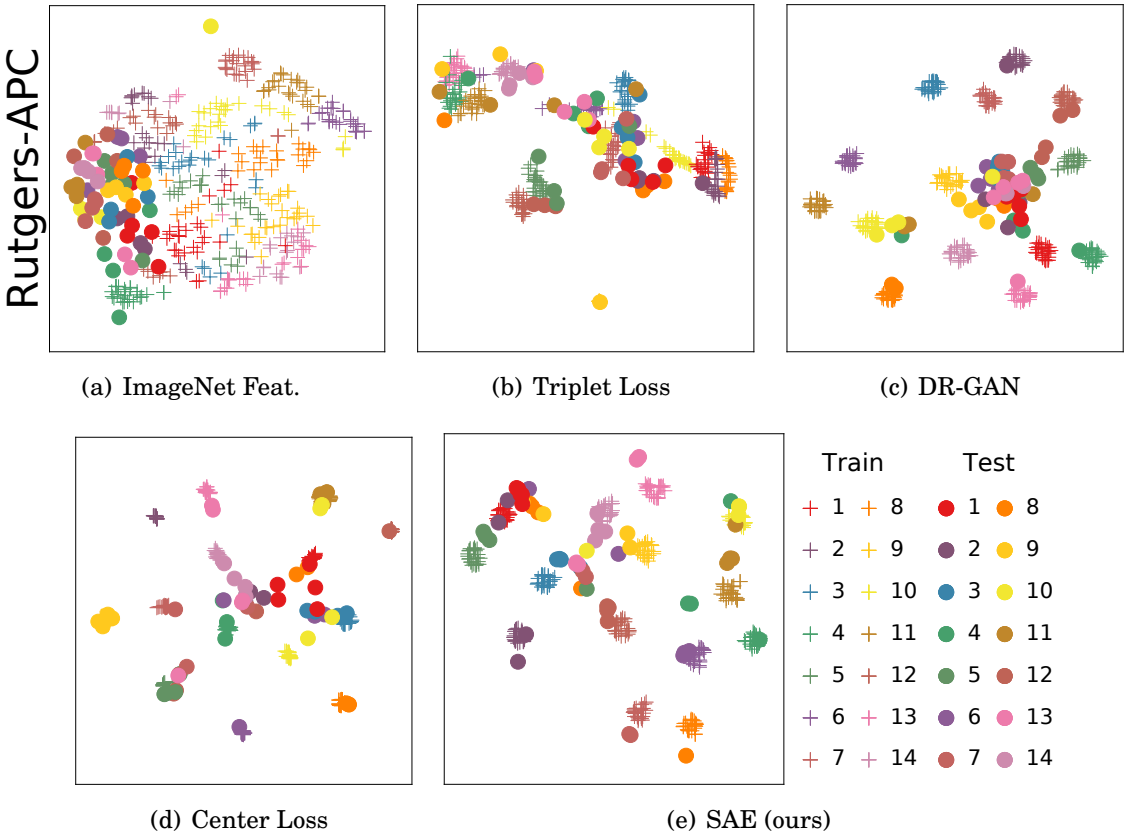


(a) ImageNet Feat. (b) Triplet Loss (c) DR-GAN

(d) Center Loss (e) S-RAE (ours)

Figure 4.6: t-SNE visualizations Tejani Dataset. Embeddings with *plus sign*(+) and *circles*(o) indicates embeddings from training and testing domain respectively. Objects are shown in Appendix A. Image better seen on the digital format.

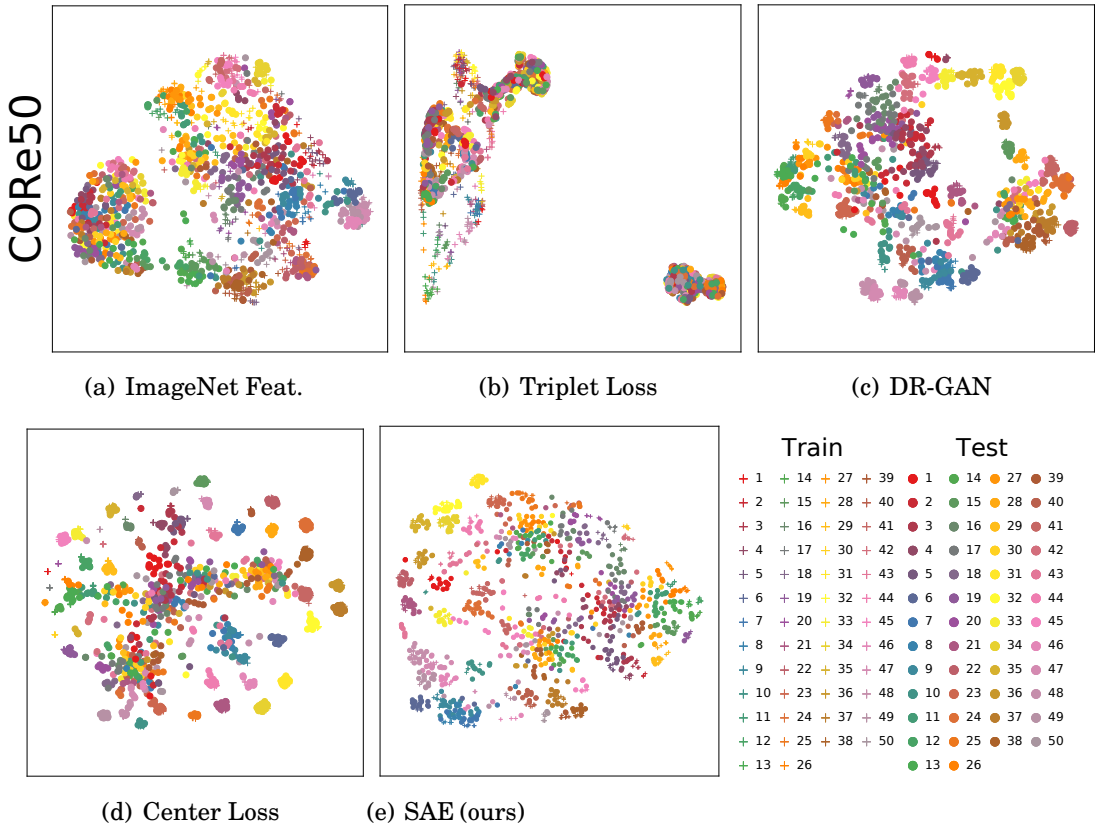
current autonomous platform. Therefore in the following chapter we address the problem of learning generalizable feature that can be use to learn to recognize new instances efficiently.



(a) ImageNet Feat. (b) Triplet Loss (c) DR-GAN

(d) Center Loss (e) SAE (ours)

Figure 4.7: t-SNE visualizations for Rutgers-APC. Embeddings with *plus sign*(+) and *circles*(o) indicates embeddings from training and testing domain respectively. Objects are shown in Appendix A. Image better seen on the digital format.



(a) ImageNet Feat. (b) Triplet Loss (c) DR-GAN

(d) Center Loss (e) SAE (ours)

Figure 4.8: t-SNE visualizations in CORe50. Embeddings with *plus sign*(+) and *circles*(o) indicates embeddings from training and testing domain respectively. Objects are shown in Appendix A. Image better seen on the digital format.

CHAPTER 5

LEARNING DISCRIMATIVE EMBEDDINGS ON-THE-FLY

In Chapter 4, we propose a model for domain generalization which was particularly useful for learning features from a single source domain. However, it struggles to learn additional instances efficiently, as it needs to be retrained to achieve state-of-the-art recognition performance. Therefore, in this chapter we propose the end-to-end learning of discriminative features by a disentangled representation of classification and image similarity that allows the learning on-the-fly of additional objects. Similarly to the previous chapter, our approach can learn from limited data, e.g. data collecting within a specific a place, and able to generalize in unseen environments and nuisances.

5.1 Introduction

In this chapter we address the problem of learning to recognize new objects *on-the-fly* which is a core component in our *in-situ* learning problem, where a model is required to process new objects as soon as they are being perceived. Over the past few years, the robotics and computer vision community have adopted Convolutional Neural Networks (CNNs) as the standard approach for addressing object recognition and localization (Huang et al., 2017b). Despite the human-level performance in recognition accuracy, CNNs are still very limited tools compared with the cognition capabilities of humans (Lake et al., 2017). One of these limitations is the ability to learn new instances efficiently, without catastrophically forgetting the previously learned ones and with the capability for generalization over the commonly encountered changes in perspective,

scale, illumination, object's pose and noise.

When using CNNs, a typical approach for learning new objects is by finetuning parts of the entire model. However, this approach relies on the assumption that the original training set is still available and requires high-end computational resources for training the ever-growing dataset efficiently, which can be unfeasible for robots with limited hardware. Therefore, we propose an alternative to the finetuning methodology by designing an architecture that:

1. Instead of just predicting labels, it learns to generate discriminative and separable features (*a.k.a.* embeddings) of an object's viewpoints by using a combination of a classification and metric learning losses on the features space.

2. Infers an object's identity efficiently by utilizing a lightweight classifier in the embeddings space, to keep inference time in the order of milliseconds and decision boundaries can be retrained efficiently when new objects are learned.

The primary goal in this chapter is to show that a single-branch, two-heads CNN can be trained effortlessly to generate discriminative and separable features which can be useful for learning new objects *on-the-fly*. We evaluate our approach on four real-world images datasets used for Robotics and Computer Vision applications: Amazon Robotics Challenge 2017 by MIT-Princeton, T-LESS, ToyBox, and CORe50 datasets.

By combining a metric and classification losses, the model learns to generate discriminative and separable embeddings in the features space $\mathbb{R}^n$, as depicted in Figure 5.1. In this embeddings space, a lightweight classifier such as k-nearest neighbors or a linear support vector machine (Song et al., 2016) can be used for predicting the identity of a query image. Furthermore, new objects can be mapped to $\mathbb{R}^n$ using the same learned model, and only the lightweight classifier has to be re-trained in order to get the predictions considering all learned objects.

We evaluate our approach in four datasets used for multi-view object recognition: T-LESS, ToyBox, Amazon Robotics Challenge 2017 by Princeton-MIT and CORe50 and we use precision as the evaluation metric.

5.2 Related Work

The most straightforward strategy for learning from a continuous stream of data using CNNs is to retrain the model (entirely or just a few layers) using the updated training

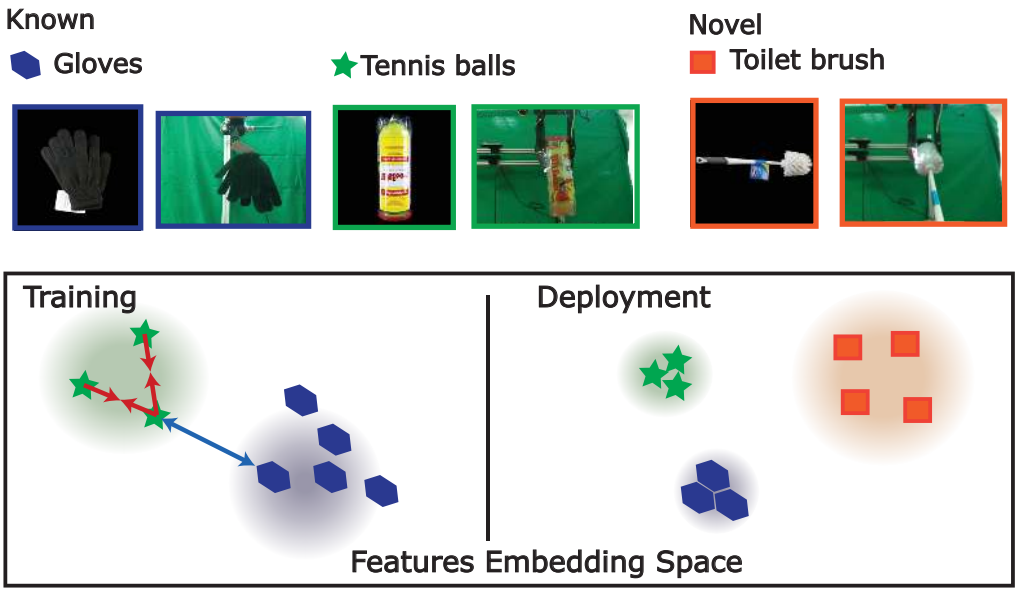


Figure 5.1: The problem we aim to solve is to learn new objects without re-training the CNN. To do so, we teach a model to maximize the separability (blue arrow) and minimize the distance of embeddings from the same instance (red arrows), in a supervised manner. During deployment, the model can produce embeddings with these properties even for objects not seen during training. In order to perform classification, we use a lightweight classifier such as k-nearest neighbors for matching training and testing embeddings.

set (Jain et al., 2014; Xiao et al., 2014; Käding et al., 2016). While these approaches offer *state-of-the-art* performance, it is an unfeasible approach for many autonomous and mobile robots with limited computational resources that are required to learn new objects efficiently. While continuous finetuning approaches have been explored (Käding et al., 2016) it is an unfeasible approach for most autonomous robotic platforms given three main limitations: 1) It assumes access to the original training set, which can easily reach the order of Terabytes (TBs) even with the storing of low-quality images. 2) It requires high-end dedicated computational resources for training the model efficiently and 3) The time required for re-training increases linearly as more items are introduced. 4) The power consumption required to re-train the model every time a new object is required to be learned, might compromise the autonomy of a mobile platform with a limited source of energy. These limitations motivate for developing models that will not require finetuning for learning new objects.

An alternative strategy to finetuning the model for label predictions, is the metric learning (*a.k.a.* similarity learning) approach, in which instead of just predicting labels, the model is trained to learn the concept of *similarity* by bringing close to each other

features from the same instance and far apart otherwise, in a feature space $\mathbb{R}^n$.

Currently, the best approaches to metric learning employ *state-of-the-art* CNNs (Song et al., 2017; Bellet et al., 2013). A pioneering model is the Siamese architecture (Hadsell et al., 2006) which utilizes the contrastive loss (Equation 5.1) defines as:

$$\mathscr{L}_{contrastive} = (1-Y)\frac{1}{2}D(X_1,X_2) + (Y)\frac{1}{2}[m - D(X_1,X_2)]_+ \tag{5.1}$$

Where X_1, X_2 are pairs of images, that can be from the same or different objects, indicated in the vector Y. A CNN f_θ with weights θ is used for mapping images to an embedding space $\mathbb{R}^n$ (Equation 5.2), where the Euclidean distance (Equation 5.3) d is used for computing the similarity between the two embeddings. Dot product, Mahalanobis distance or even a trainable metric as in (Xing et al., 2002), has been proposed as an alternatives to Equation 5.3. Finally, the $[\cdot]_+$ operator denotes the hinge function equivalent to $max(0,\cdot)$ function.

$$D(X_a,X_b) = d(f_\theta(X_a), f_\theta(X_b)) \tag{5.2}$$

$$d(x_a,x_b) = \frac{1}{2}\|x_a - x_b\|_2^2 \tag{5.3}$$

While the siamese architecture has been used successfully for one-shot learning (Koch et al., 2015), dimensionality reduction (Hadsell et al., 2006), image classification (Kumar et al., 2015), and cross-domain adaptation (Zeng et al., 2018) with competitive results, it is unable to learn the concepts of similarity and dissimilarity at the same time, since the pairs are either from the same or different object. We show more details about its limitations in the experiments section.

This limitation is addressed in the triplet architecture (Schroff et al., 2015), which encourages a relative distance constraint between similar and dissimilar images simultaneously in the triplet loss (Equation 5.4).

$$\mathscr{L}_{triplet} = \left[D(X_a,X_p) - D(X_a,X_n) + m\right]_+ \tag{5.4}$$

The triplet architecture became the most widely used approach for metric learning (Zhe et al., 2018), with two main limitations:

1. Sampling a useful set of the triplets in Equation. 5.4 is a non-trivial and crucial task (Schroff et al., 2015), given that f_θ quickly learns to correctly map most trivial triplets, it leaves a large fraction of all triplets uninformative and makes sophisticated mining of triplets a necessary step for training (Wu et al., 2017).

2. The loss is defined in terms of small groups of images in the mini-batch and does not consider a global structure of the training set, which might lead to sub-optimal solutions depending on the mini-batch size.

Thus, many variants of the triplet architecture aim to address these issues. Concerning mini-batch formation, a first approach for hard mining triplets is the Lifted Structure (Song et al., 2016) approach, which consists of incorporating on-line hard negative mining by comparing each positive example against all negative examples in the training mini-batch. The Quadruplet (Chen et al., 2017) architecture incorporates an additional negative example in the mini-batch aiming to facilitate the clustering of negative examples. Quintuple (Huang et al., 2016a) and N-pair (Sohn, 2016) models are extensions of this idea.

For selecting the best triplets, (Kumar et al., 2017) proposes a smart mining trainable module that forms triplets from a pool composed of semi-hard positive and negative samples. Similarly, (Huang et al., 2016b) proposes a trainable module named position-dependant deep metric (PDDM) that scores the hardest negative example within the mini-batch, based on relative and global distances within the batch. More recently, (Hermans et al., 2017) proposed Batch Hard forming, where the core idea is to form batches by randomly sampling P classes (i.e., instances) and then selecting the hardest positive and the hardest negative samples within the batch. An orthogonal approach for speeding up the training is angular loss (Wang et al., 2017b) where authors propose an additional angular loss term that constrains the upper bound angle in each triplet triangle.

For addressing the lack of global structure in the metric space, (Gecer et al., 2017) proposes to combine sample-based methods (such as triplets or softmax) with set-based methods (such SMVs). This approach is trained with a max-margin loss, which improves the separability of the embeddings by maximizing the possible inter-class margin by using support vector machines (SVMs). The main limitation, having to train the model set-based offline and the sample-based step online.

In (Kumar et al., 2015), authors propose a global loss that tries to minimize the variance within each embedding distribution and maximize the mean value of the

distances between non-matching pairs. The drawback of this method is requiring the complete training set loaded in memory, in order to estimate variance and means in each class.

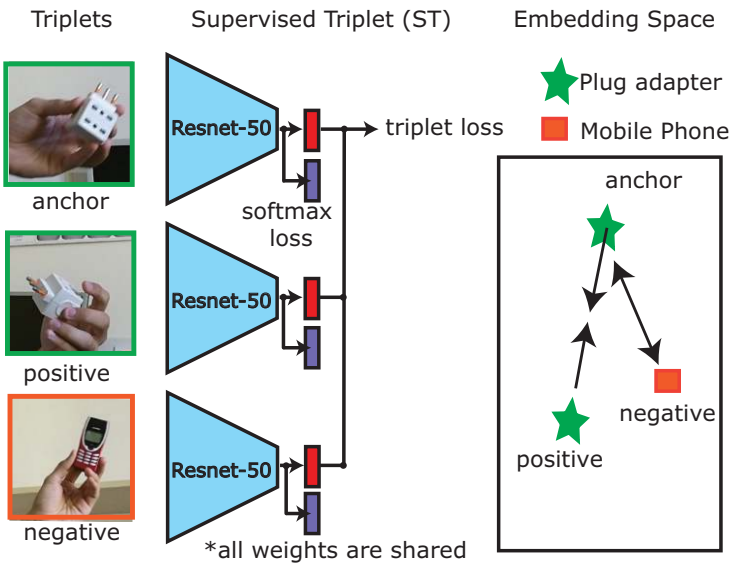
An orthogonal approach to mini-batches with tuples (pairs, triplets, n-pairs, etc.) is the Facility Location (Song et al., 2017) approach, where instead of training the model based in the similarity between pairs or triplets, they learn directly to cluster classes by mapping each example i to its nearest point from a chosen set of landmarks defined beforehand. While this approach achieved *state-of-the-art* for image retrieval tasks, the number and location of medoids classes has to be fixed in order to precompute a ground facilities, which makes it an unsuitable approach for the learning of novel objects *on-the-fly*.

Another approach beyond the triplets is boosting independent embeddings robustly (BIER) (Opitz et al., 2018), where dot product is used as a comparison metric and the embedding layer is divided into multiple non overlapping groups, which are treated as an online gradient boosting problem. This approach is particularly beneficial for large embedding sizes. More recently, Directional statistics (Zhe et al., 2018) proposes a more suitable embedding representation based on L2-normalization and the Mises-Fisher distribution. The resulting embedding is contained in a hyper-spherical embedding space, using cosine similarity and trained with a new loss function named von Mises-Fishes (vMF) loss.

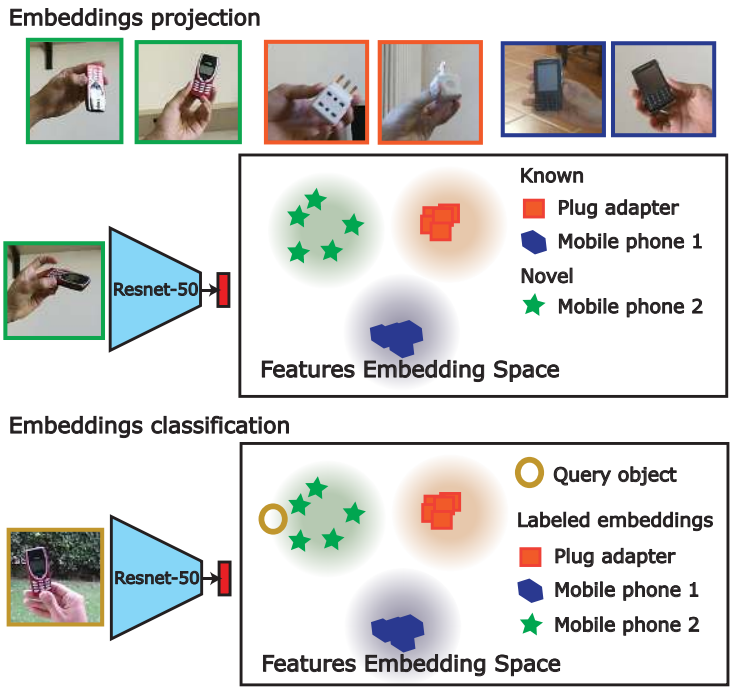
More recently, center loss (Wen et al., 2016) proposes a model with the same goal of minimizing the intra-class distance of the embeddings, by learning a center for each class and penalizing the distances between the embeddings and their corresponding class centers. However, instead of recurring to cumbersome sampling techniques, it utilizes the cross-entropy loss to aid the optimization process. In the search for a model with higher discriminable embeddings in the features space, (He et al., 2018) combines the triplet and center loss for multi-view 3D object retrieval.

Our Supervised Triplet Network (s-triplet) follows the same idea from center loss, by training a CNN with triplets of images and using a soft margin triplet loss, and utilizing the softmax loss computed from each image and its label.

The representations in the embedding space has been used for person re-identification (Schroff et al., 2015; Hermans et al., 2017), clustering and retrieval (Song et al., 2016, 2017; Opitz et al., 2018; Sohn, 2016; He et al., 2018) and object recognition (Koch et al., 2015; Milan et al., 2017; Zeng et al., 2018). To do so, a classifier can directly be constructed in the feature space, being k-nearest neighbors and support vector machines, the most



(a) Training



(b) Inference

Figure 5.2: Proposed Object Recognition *on-the-fly* framework. Our Supervised Triplet Network (s-triplet) (a). For performing classification, the model will project an image's embedding close to the most similar labeled embeddings, for which a light-weight classifier, such as k-nearest neighbors, can be used to estimate the unknown object.

popular options.

5.3 Proposed Method

Our approach consists of dividing the recognition problem into two stages: Embeddings generation and classification in the embeddings space. We use a CNN to generate separable and discriminant features from an object's viewpoints in a generic fashion, this is, the embeddings generated does not contain information about the identity of the object. This allows for the model to be applied to unseen objects. With separable and discriminant features projected in an embedding space, a lightweight classifier such as the k-nearest neighbors or a linear support vector machine, are used for computing the predictions given a query image. When a new object is needed to be learned, only the lightweight classifier is re-trained in order to update its decision boundaries.

5.3.1 A Loss Function for Learning Discriminative Embeddings

Thus, we propose a framework that combines the benefits of metric learning and label prediction approaches by utilizing the softmax loss in combination with the triplet loss as regularizer, as shown in Fig. 5.2(a). The loss function in our s-triplet is defined by:

$$\mathscr{L}_{striplet} = \mathscr{L}_{softmax} + \lambda \cdot \mathscr{L}_{triplet} \tag{5.5}$$

$$\mathscr{L}_{triplet} = \left[d(\hat{x}_a, \hat{x}_p) - d(\hat{x}_a, \hat{x}_n) + m \right]_+ \tag{5.4}$$

$$\mathscr{L}_{softmax} = -\sum y log(q(x)) \tag{5.6}$$

Each loss term works as follows:

As studied in (Wen et al., 2016) and (He et al., 2018), using a fully supervised loss such as the softmax loss can be beneficial in metric learning models, where sampling relevant examples into the mini-batches is still an open research problem. The idea behind of using the softmax loss with the logits x from each training example, is to generating earlier manifolds efficiently, this is, by utilizing all the images in the mini-batch as oppose to the very expensive time required by triplets (Song et al., 2017).

For the triplet loss, the mini-batches consist of a combination of images X_{a_i} selected randomly which are used in the softmax loss and triplets with the anchors images X_{a_i} as the previously objects selected randomly. The positive X_p and negative X_n examples of the anchor images are selected randomly. We then compute the Euclidean distance d, between the embeddings from anchor, and positive and negative examples.

Similar to center loss (Wen et al., 2016), it is possible to learn a more discriminative visual representations when the softmax loss is constrained by penalizing the similarity of the features in the embeddings space $\mathbb{R}^n$. To do so, we use Euclidean distance for comparing the similarity of two embeddings. As oppose to (Wen et al., 2016), we use the triplet loss, which brings embeddings from the same instance together and away otherwise without the need of computing centroids from every instance, in every training step.

As studied by (He et al., 2018), the metric loss has to be attenuated by a factor λ in order to give priority to the manifolds generated by the softmax loss, this is the only additional hyperparameter in our model respect to the Triplet Network (Schroff et al., 2015).

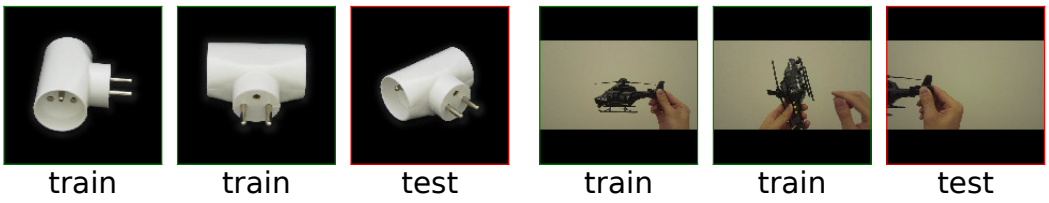
In Figure 5.2, we indicate how the softmax and triplet losses uses the features from the last convolutional layer in the backbone CNN. Additionally each of this representations are separated in order to reduce the bias from the known object in the embeddings generation, as shown in detail in the ablation studies.

So far, the CNN model can produce cluster-like embeddings from an object's viewpoints, which can be applied to objects not seen during training. However, an additional classifier is required in order to compute the prediction probabilities considering all the objects learned. Linear support vector machines and the k-nearest neighbors search are common choices for image retrieval, face identification and object recognition.

5.3.2 Feature Extraction and Embeddings Matching

We follow the methodology for projecting images into a common features embeddings space and computing the prediction probabilities presented in Chaper 4, consisting of:

1. First, the model is compacted as shown in Figure 5.2(b). Since all the weights in the model are shared, we can remove the two additional siblings and leave the model with a single branch. Additionally, we also remove the fully connected layer that was used for the softmax loss.



(a) T-LESS (b) ToyBox

(c) Amazon Robotics Challenge (d) CORe50

Figure 5.3: Object Recognition Datasets. In T-LESS (Hodaň et al., 2017), the model has to recognize unseen viewpoints. In Toybox (Wang et al., 2017c) we used the hodgepodge videos for training and translations across the three axes for testing. We use ARC as in (Zeng et al., 2018). We use training and testing scenes as proposed by (Lomonaco and Maltoni, 2017)

2. For learning a new object, we project the training images to the feature embeddings space $\mathbb{R}^n$ and associate each data point with its respective label.

3. We retrain the lightweight classifier for making predictions by taking into account the added embeddings in $\mathbb{R}^n$.

4. For recognizing the object, we project a query image to $\mathbb{R}^n$, and the classifier will compute the prediction probabilities.

The training examples are stored as embeddings, so there is no need for saving image files. Any CNN architecture can be trained with Equation 5.5 for producing the embeddings. Similarly, any classifier can be used for predicting the decision boundaries and probabilities in $\mathbb{R}^n$. In the next section, we describe how the selection of the classifiers affects performance and computing times.

5.4 Experimental Results

5.4.1 Datasets

We selected four datasets that depict objects from an egocentric view, as would be seen from a robot's perspective for most mobile robots and manipulators. We show an example of each dataset in Figure 5.3 and the remaining objects in Appendix A. Apart from the Amazon Robotics Challenges by Princeton-MIT, where an actual robot took images, we selected other datasets that present challenging recognition scenarios that a robot might face in real conditions. The selected datasets are :

T-LESS (Hodaň et al., 2017): This datasets contains 30 objects with no relevant texture, objects are part from other. For this chapter, we randomly sampled the 10% from the training set, to be used as the new training set and we use the remaining 90% as testing set. The motivation behind is to evaluate how well the model can infer if an unknown viewpoint belongs to a given instance. This results in 348 images for training and 37,027 images for testing.

ToyBox (Wang et al., 2017c): It is composed by 360 toys manipulated by a person. Toybox allowed us to evaluate how well the model scales by learning 120 novel objects *on-the-fly*. We utilize the hodpodge videos for training and the translations and rotations across x,y,z axis for testing. The testing set also depicts new conditions such as changes in scale, partial views and occlusions by different hands. We use 24,667 images for training and 69,777 test images as in the previous chapter.

Amazon Robotics Challenge (Zeng et al., 2018): This dataset was collected from the Amazon Robotics Challenge 2017, consists of 60 objects collected by the MIT-Princeton robot and contains training images from two different domains: Synthetic images provided by Amazon and real images captured by an RGB camera showing the objects being grasped by the robot. For this particular dataset, we use the methodology proposed by (Zeng et al., 2018) consisting of restricting the number of instances in the embeddings space. This is, every testing image is compared against 20 possible instances, as opposed to the total of 60 instances. This is part of the rules of the competitions and we followed them in order to make our results comparable with the ones from (Zeng et al., 2018). In the next chapter we consider the general case of comparing a query image against all possible objects. We use the total 10,042 images for training and 562 images for testing.

CORe50 (Lomonaco and Maltoni, 2017): Originally proposed for continuous learning, this dataset shows 50 objects across eleven environments and allows us to test the generalization capabilities not only to unseen object's poses but also new environments.

We utilize the standard testing set composed by scene 3,7 and 11, and the remaining scenes for training the model. This results in using 119,894 images for training and 44,972 images for testing.

5.4.2 Baselines

As the first baseline, we selected ResNet-50 trained with ImageNet, for generating embeddings. Since ResNet-50 does not have intermediate fully connected layers, we follow the methodology proposed by (Zeng et al., 2018) which consists of taking the feature map after the average pooling layer and flatten it to a 1-D vector with 2048 elements (this dimension resulted from using a resolution of 224×224 pixels). This baseline allowed us to explore if a model trained with a large dataset can produce discriminative embeddings for multi-view object recognition.

As the second baseline we selected the Siamese Network since it exhibited usefulness in (Zeng et al., 2018) for recognizing novel object *on-the-fly*. We applied hard-negative mining as proposed in (Schroff et al., 2015) and (Song et al., 2016). The hard-negative mining consisted of taking the 20% more difficult examples in the mini-batch and feeding them in the next mini-bath. The sibling CNNs were pre-trained with ImageNet as we are using a few examples per object.

Our last baseline is the center loss, since it utilizes a supervised loss for generating discriminative embeddings as in our method. We selected the hyperparameters $\lambda = 0.1$ and $\alpha = 0.005$ in the loss function (Wen et al., 2016) and utilized the PyTorch implementation from (Zhou, 2018).

For the siamese and s-triplet (our approach) we used a soft margin, this is $m = 0$ in Equation 5.4, as recommended by (Hermans et al., 2017). For the Amazon Robotics Challenge dataset, we also compared our model against the winning team (Zeng et al., 2018) and their baseline Siamese Network (Hadsell et al., 2006). We did not select (Zeng et al., 2018) as a baseline for the other datasets as it is not detailed by the authors how to select the image pairs from a single or multiple (more than two) cross-domain datasets.

5.4.3 Implementation Details

We selected ResNet-50 (He et al., 2016) as the backbone CNN for generating features and k-nearest neighbors with $k = 5$ as the classifier in order to make our model comparable with the Amazon Robotics Challenge 2017 winner (Zeng et al., 2018), who uses the same backbone CNN. As we show in Figure 5.2(b) ,a fully connected layer generates the

embeddings with a dimension of 128 elements. Selecting the embedding size has been studied by (Schroff et al., 2015) and (Vassileios Balntas and Mikolajczyk, 2016) in the context of face recognition and image retrieval, choosing a dimension of 128 elements led to both faster inference and comparable accuracy compared to higher dimensions. We developed our models using PyTorch 1.0, trained with GTX 1080Ti GPUs with 11GB, code is available in (Lagunes-Fortiz, 2019a).

5.4.4 k-Nearest Neighbors vs Label Prediction

We first compare the recognition performance from a model trained with a metric learning loss against a label predictions approach. While all the models uses the same amount of training examples, only the CNN (trained only with a softmax loss) and s-triplet (ours) utilizes the labels information, in the siamese network a vector indicates if two images corresponds to the same or different object, in the triplet network a similar and a negative images are included in each mini-batch.

The predictions for the CNN (trained purely with the softmax loss) model were obtained from the fully connected layer which was trained to do so, for the rest of the model, the predictions were obtained by a k-nearest neighbors search, with $k = 5$.

From Table 5.1, there is an evident gap in performance between the CNN (trained only with a softmax loss) model and the metric learning approaches trained similarity loss. The gap is more significant in domain-adaptation scenarios such as ARC and CORe50. This decrease in performance is an inherent drawback in current metric learning approaches based purely on *image similarity* since testing images depicting objects in a different background and illumination conditions are not similar enough to the training examples which led to a wrong mapping into the correct instance manifold. On the other hand, a CNN trained with a fully supervised loss dealt better with the domain generalization scenario.

Our model achieved a consistent higher performance compared to the CNN trained with softmax loss by around 3% in average, as we show in the t-SNE visualizations in the following section, this slight increase in performance is partially explained due to the higher discriminability in the features produced by the model.

5.4.5 Recognizing Novel Objects

Here we test the capabilities of each model for recognizing novel objects *on-the-fly*, that is, without re-training or finetuning the backbone CNN model. We followed the methodology

Table 5.1: Models trained for label prediction vs. metric learning approaches % Accuracy top-1 recognition

Method	TLESS	ToyBox	ARC	CORe50
CNN (Softmax loss)	97.31 ± 0.17	74.96 ± 0.08	92.31 ± 0.13	92.61 ± 0.26
CNN (ImageNet)	34.81	10.13	27.2	24.89
Siamese CNN	52.21 ± 2.34	38.31 ± 3.41	49.23 ± 3.56	39.22 ± 3.56
Triplet CNN	93.65 ± 1.49	56.98 ± 1.32	75.17 ± 1.49	74.14 ± 1.83
S-Triplet (ours)	**98.59 ± 0.42**	**76.58 ± 0.79**	**96.09 ± 0.21**	**94.06 ± 0.41**

proposed by (Zeng et al., 2018) which consists of splitting each dataset into a *novel* and *known* sets, two-thirds of the objects are used for training the model and the remaining third is used for recognition of new instances. The objects are selected randomly, for the ARC dataset, we used the same splits as in (Zeng et al., 2018), for T-LESS, CORe50 and ToyBox the splits are located in Appendix A. Every model was trained three times, using the same train/test split and we show the mean and standard error to show the variance of each model. Results are shown in Table 5.2.

For the T-LESS dataset, we can infer that all approaches, but the siamese network, were able to map correctly unseen viewpoints of each object since training and testing images have the same background and illumination conditions. The center loss was the model with the closest performance to our approach, with a higher recognition performance for T-LESS but struggling in the remaining datasets with increased complexity. Our approach achieved an average 4% higher performance for known objects, 5% for novel ones and 7% higher for the general case of combined known and novel objects, compared with the center loss.

In Table 5.2 we also include the comparison of our results with the Amazon Robotics Challenge 2017 winner team (Zeng et al., 2018), our model has a comparable performance, only 1% less for the cases of novel and combined sets, and 3% higher for recognizing known object. Compared to (Zeng et al., 2018), our model utilizes only one CNN, which translates in half of the parameters to be trained and saved. Additionally, our approach does not require a careful selection of the examples for learning the concept of similarity, since it learns such a concept in combination with a softmax loss, which is trained efficiently for generating manifolds. Efficient optimization means that our model does not require any cumbersome mining technique nor distance metrics, which makes implementation easier across current recognition datasets.

Table 5.2: Learning novel objects *on-the-fly* % Accuracy top-1 recognition

Objects	ImageNet Features	Contrastive Loss (Hadsell et al., 2006)	Center Loss	K-net + N-net (Zeng et al., 2018)	**Supervised Triplet Loss (ours)**
(h) T-LESS					
Known	72.93	45.23 ± 3.41	99.38 ± 0.79	-	**99.41 ± 0.71**
Novel	85.45	38.91 ± 2.65	**99.71 ± 0.84**	-	97.51 ± 0.59
Known & Novel	75.81	32.19 ± 3.17	**96.54 ± 0.66**	-	96.26 ± 0.79
(c) ToyBox*					
Known	48.83	45.31 ± 2.58	74.76 ± 0.43	-	**81.52 ± 1.32**
Novel	51.18	39.76 ± 3.12	76.48 ± 0.59	-	**80.52 ± 1.25**
Known & Novel	49.46	32.56 ± 3.43	67.67 ± 0.46	-	**73.23 ± 1.29**
(d) ARC-MIT					
Known	27.20	42.23 ± 2.30	86.74 ± 0.53	93.6	**96.09 ± 0.88**
Novel	52.61	52.31 ± 2.23	64.21 ± 0.89	**77.5**	74.21 ± 0.71
Known & Novel	35.01	44.12 ± 3.12	69.54 ± 0.77	**88.6**	87.53 ± 0.62
(e) Core50					
Known	59.95	38.41 ± 2.18	93.36 ± 0.69	-	**95.31 ± 1.32**
Novel	66.82	17.65 ± 2.43	84.55 ± 0.73	-	**89.03 ± 1.26**
Known & Novel	54.12	32.29 ± 2.87	83.83 ± 0.68	-	**87.23 ± 1.25**
Average					
Known	52.22	42.80 ± 2.61	88.56 ± 0.61	-	**93.08 ± 1.05**
Novel	64.02	37.15 ± 2.60	81.23 ± 0.76	-	**85.31 ± 0.95**
Known & Novel	53.06	35.29 ± 3.14	79.40 ± 0.68	-	**86.06 ± 0.98**

5.4.6 Ablation Studies

As ablation studies, we first evaluated the performance in two variants of the combined triplet and softmax losses. The first variant consist of a sequential combination of the losses as shown in Figure 5.4(a). This *single-branch* configuration is inspired by the center loss, where first, features are constricted by the metric loss and then the softmax loss is utilized and the end of the CNN. We evaluated a second approach, where the representations from the metric and softmax losses are separated. This second approach, which we refer as the *two-branch* model (Figure 5.4(b)), aims to reduce the bias from the features learned from the softmax loss and have more general representation in the embeddings.

We also evaluated the performance when the model is purely trained with a softmax loss, as it is commonly performed in CNNs and also a model purely train with the triplet Loss. We performed the recommended semi-hard mining for selecting relevant triplets as performed in (Schroff et al., 2015).

The results are presented in Figures 5.4(c) - 5.5(e). We show how the variant with separated layers for the metric and softmax losses, achieves a better performance for novel and combines known/novel objects recognition, at expenses of a slight reduction in performance (around 2%) for the detecting known objects. Additionally, we show how the combination of the two losses leads to a higher recognition performance than the models

with separated losses.

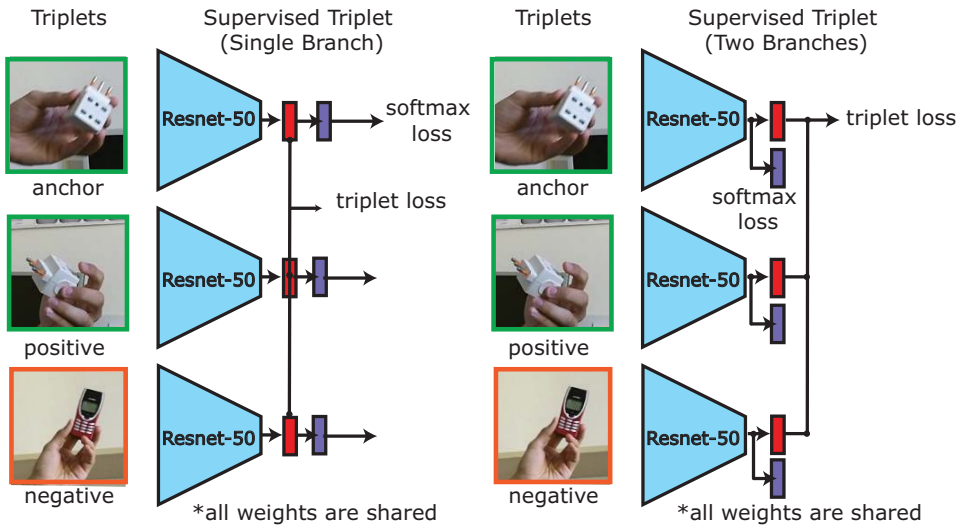
In Figure 5.5(f) we showed the effect of different lambda values, which balances the effect of the triplet loss. We followed the methodology proposed by (He et al., 2018) for combining deep metric and supervised losses, consisting of relaxing the triplet loss by a factor of λ and varying the value from 1×10^{-4} to 1, in steps of one order of magnitude. We use the 80% training set for finding the best λ and evaluating on the remaining 20%. We then use the best λ with full training splits for the rest of the experiments (Figure 5.5(f)).

Finally, in Figure 5.5(g) we explore how the performance is affected by utilizing logistic regression, a linear support vector machine (SVM) and k-nearest neighbors in the features produced by our model We found that the SVM and logistic regression achieve a faster inferences times and a consistent slight recognition performance than k-nearest neighbors with $k = 5$, at expenses of a slightly higher computational times for training the classifiers.

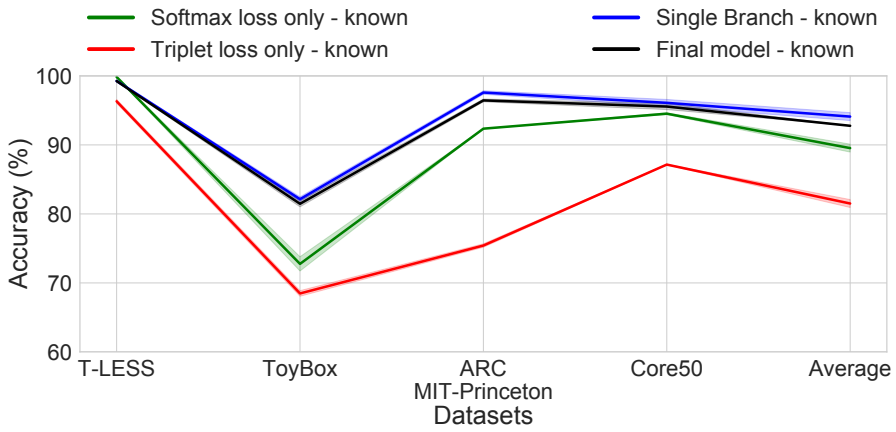
Finally, to further verify our intuition of a successful generation of discriminative features for both known and novel objects, we present t-SNE visualizations for T-LESS (Figure 5.6) and CORe50 dataset (Figure 5.8). The objects from the legends are shown in the Appendix A.

5.5 Summary

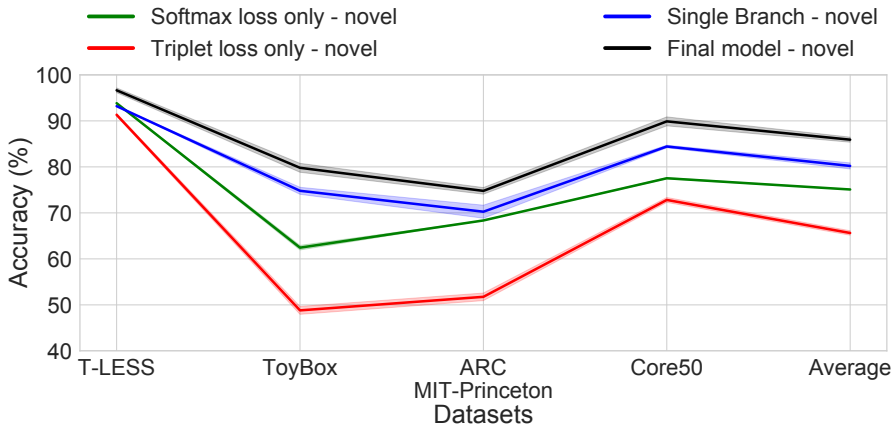
In this chapter we address the problem of learning to recognize new objects without the need of re-training the backbone CNN. We propose a new architecture that: 1) Instead of predicting labels, it learns to generate discriminative and separable embeddings of an object's viewpoints by using a supervised triplet loss, which is easier to implement than current smart mining techniques and the trained model can be applied to unseen objects. 2) Infers an object's identity efficiently by utilizing a lightweight classifier in the features embedding space, this keeps the inference time in the order of milliseconds and can be retrained efficiently when new objects are learned. We evaluate our approach on four real-world images datasets used for robotics and computer vision applications: Amazon Robotics Challenge 2017 by MIT-Princeton, T-LESS, ToyBoX, and CORe50 datasets. Our s-triplet, has a much closer performance to the finetuned models across all datasets, it achieves an average 4% higher performance for known objects, 5% for known ones and 7% higher for the general case of combined known and novel objects, compared with center loss, the closest baseline, and a much higher performance compared to semi-



(a) Single - Branch (b) Two - Branches

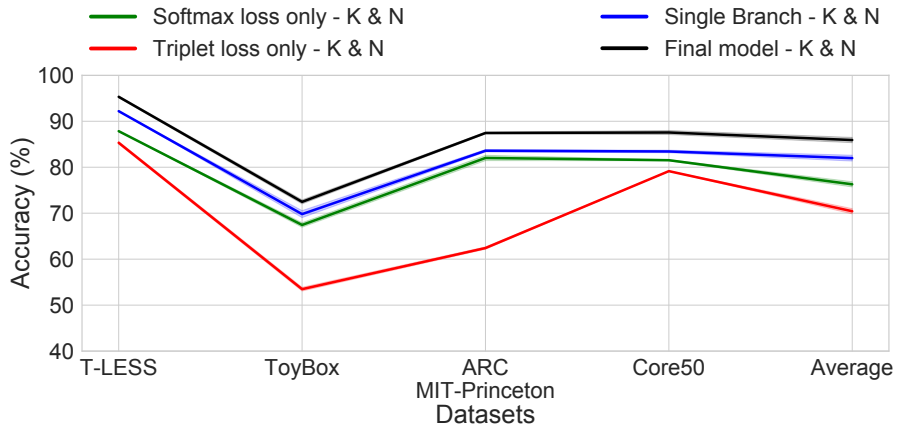


(c) Known

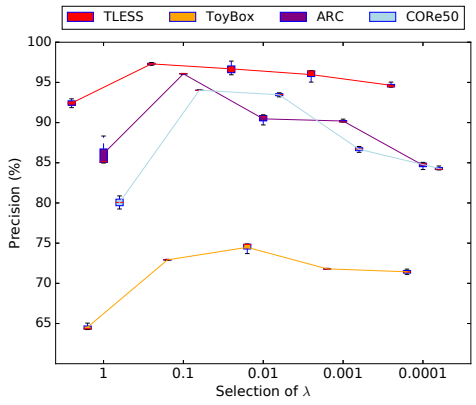


(d) Novel

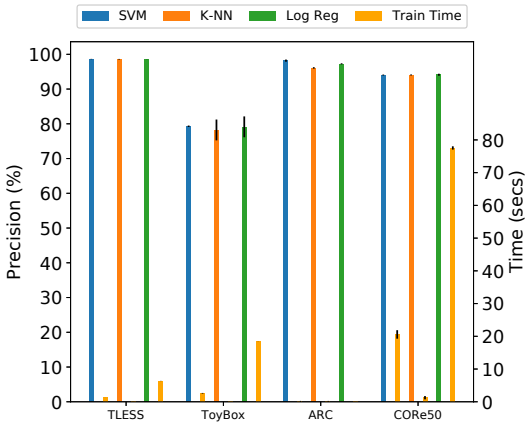
Figure 5.4: Ablation studies



(e) Know and Novel



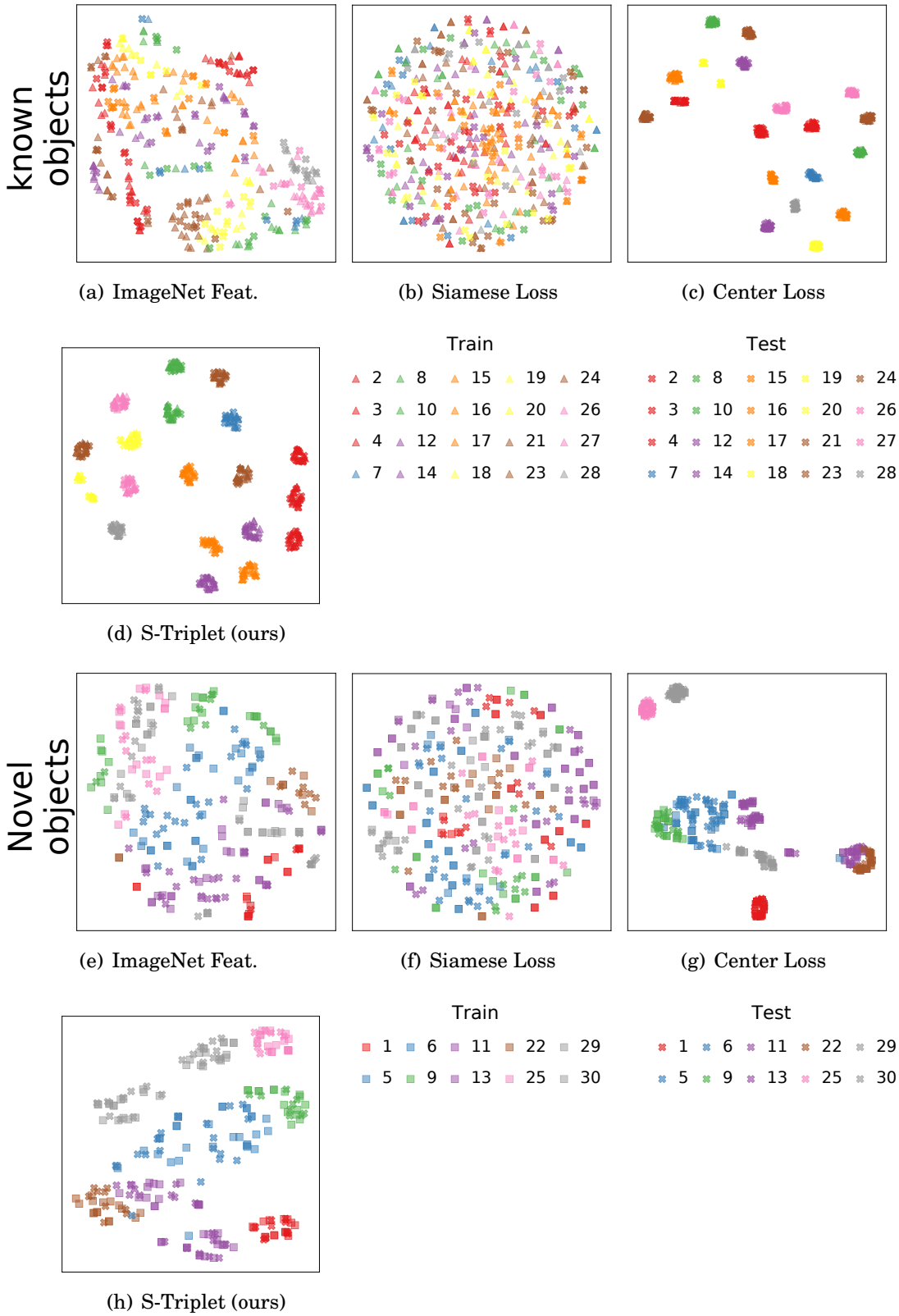
(f) Lambda Selection



(g) Classifiers

Figure 5.4: Ablation studies (continued)

supervised approaches such as the triplet and contrastive losses. This suggests that the strategy of combining a fully supervised and metric learning losses resulted in a model that generates embeddings for learning new objects *on-the-fly*. Combining these losses involved choosing a factor λ which, starting with a value of 0.1 and decreasing to values with one order of magnitude less resulted in a useful strategy across all datasets.



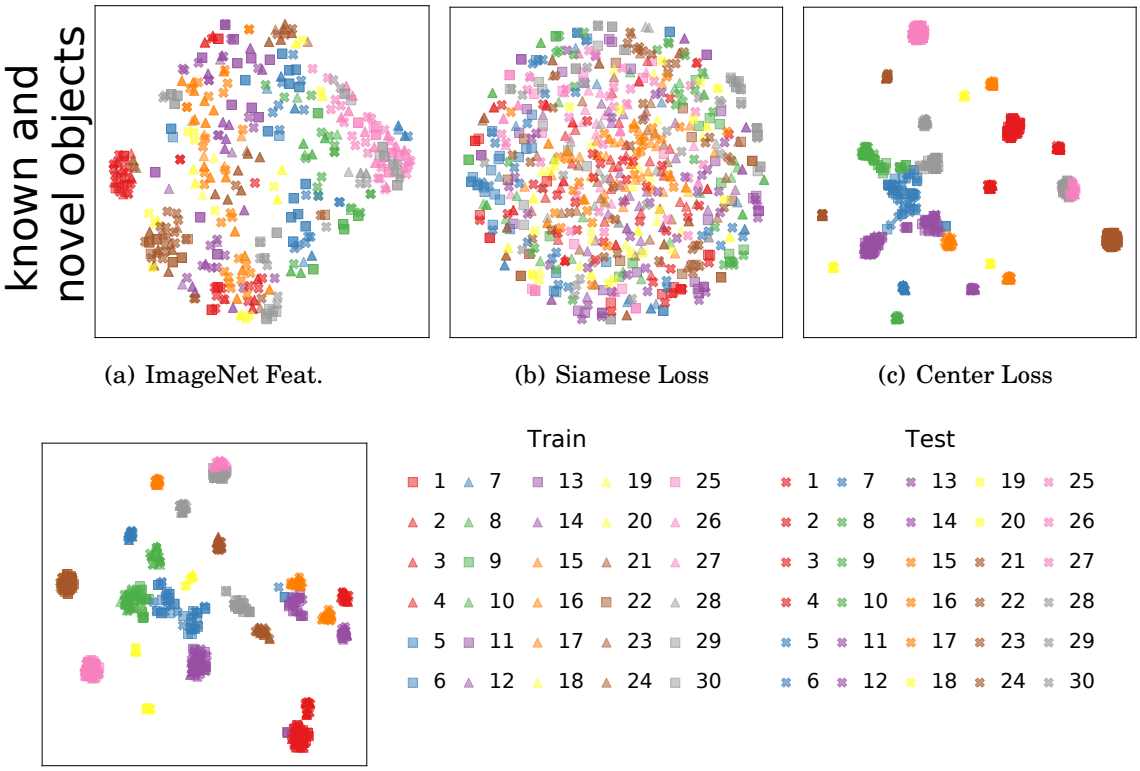
(a) ImageNet Feat. (b) Siamese Loss (c) Center Loss

(d) S-Triplet (ours)

(e) ImageNet Feat. (f) Siamese Loss (g) Center Loss

(h) S-Triplet (ours)

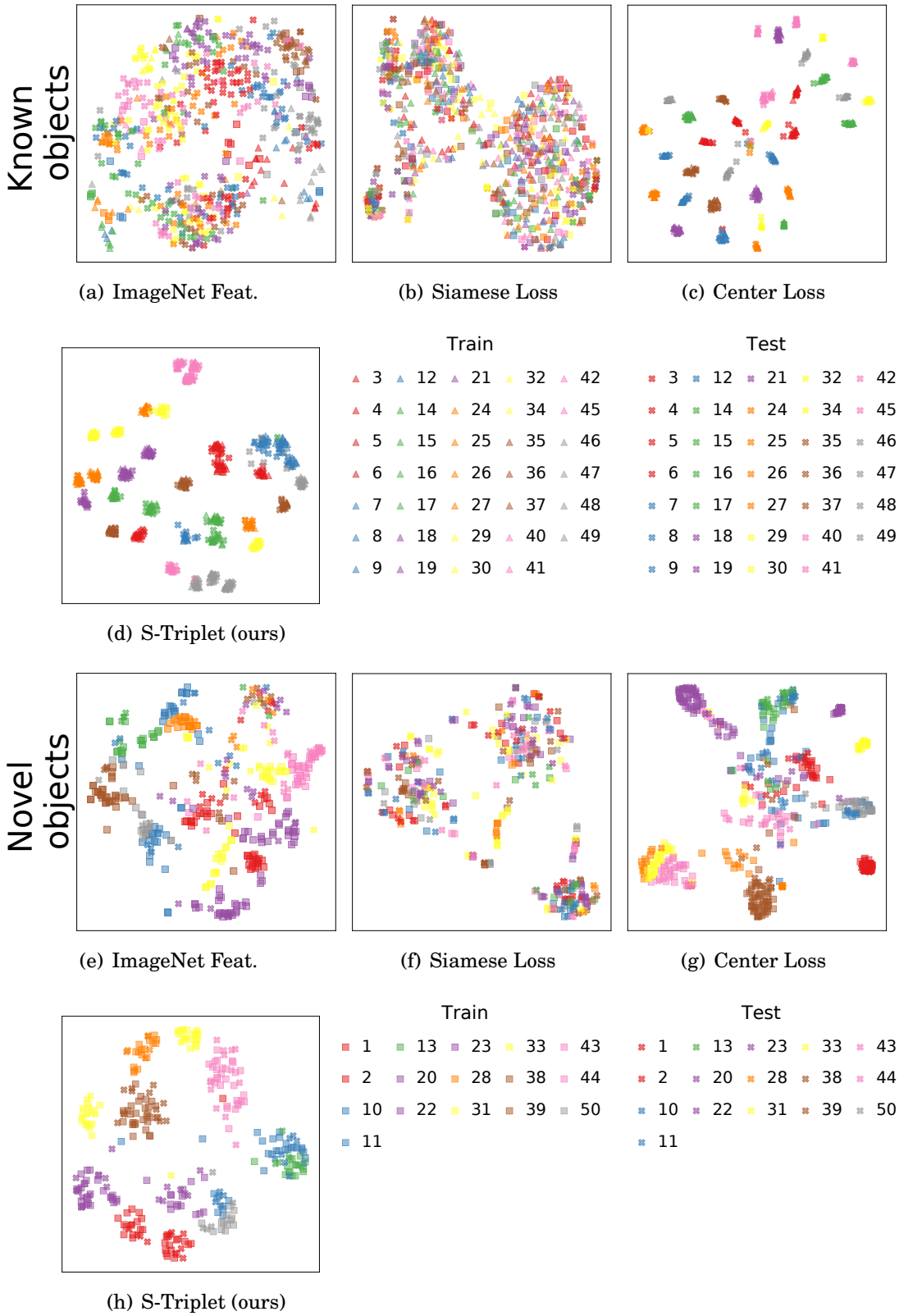
Figure 5.5: t-SNE visualizations in TLESS



(a) ImageNet Feat. (b) Siamese Loss (c) Center Loss

(d) S-Triplet (ours)

Figure 5.6: t-SNE visualizations in TLESS (continuation).



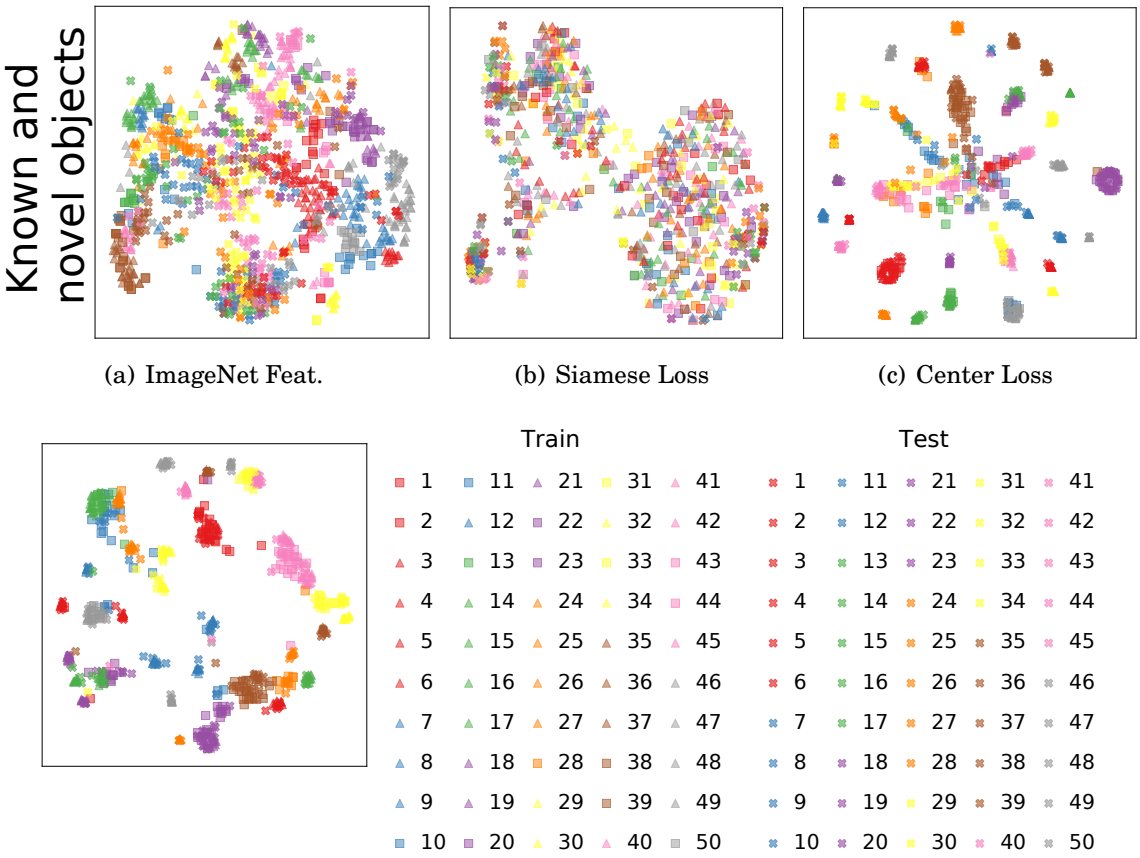
(a) ImageNet Feat. (b) Siamese Loss (c) Center Loss

(d) S-Triplet (ours)

(e) ImageNet Feat. (f) Siamese Loss (g) Center Loss

(h) S-Triplet (ours)

Figure 5.7: t-SNE visualizations in CORe50



(a) ImageNet Feat. (b) Siamese Loss (c) Center Loss

(d) S-Triplet (ours)

Figure 5.8: t-SNE visualizations in CORe50 (continuation).

CHAPTER 6

CENTROIDS TRIPLET NETWORK AND TEMPORALLY-CONSISTENT EMBEDDINGS FOR IN-SITU OBJECT RECOGNITION

This chapter aims to bring the learnings from previous chapters towards a system that does in-situ learning and deployment. We specially built on the findings from Chapters 3 and 5, for proposing a model that generates discriminative features on-the-fly, and a temporal filter that considers the consistency that must exist between neighboring video frames. The temporal consistency refers to embeddings from neighboring video frames need be close to each other in the features space, as oppose with big separations that might lead to misclassification. Additionally, we present a dataset that presents training and testing images collected in a truly in-situ scenario, and allows a systematic benchmarking of in-situ learning.

6.1 Introduction

This chapter addresses the *in-situ* learning of objects, characterized by a learner that can process new objects *on-the-fly*, using a few examples collected within the natural or common environment where objects are used or placed, and that can generalize to unseen environments.

In Chapter 5, we propose the Supervised Triplet Network (S-Triplet), a model that produces discriminative features of unseen objects, which utilizes a combined metric

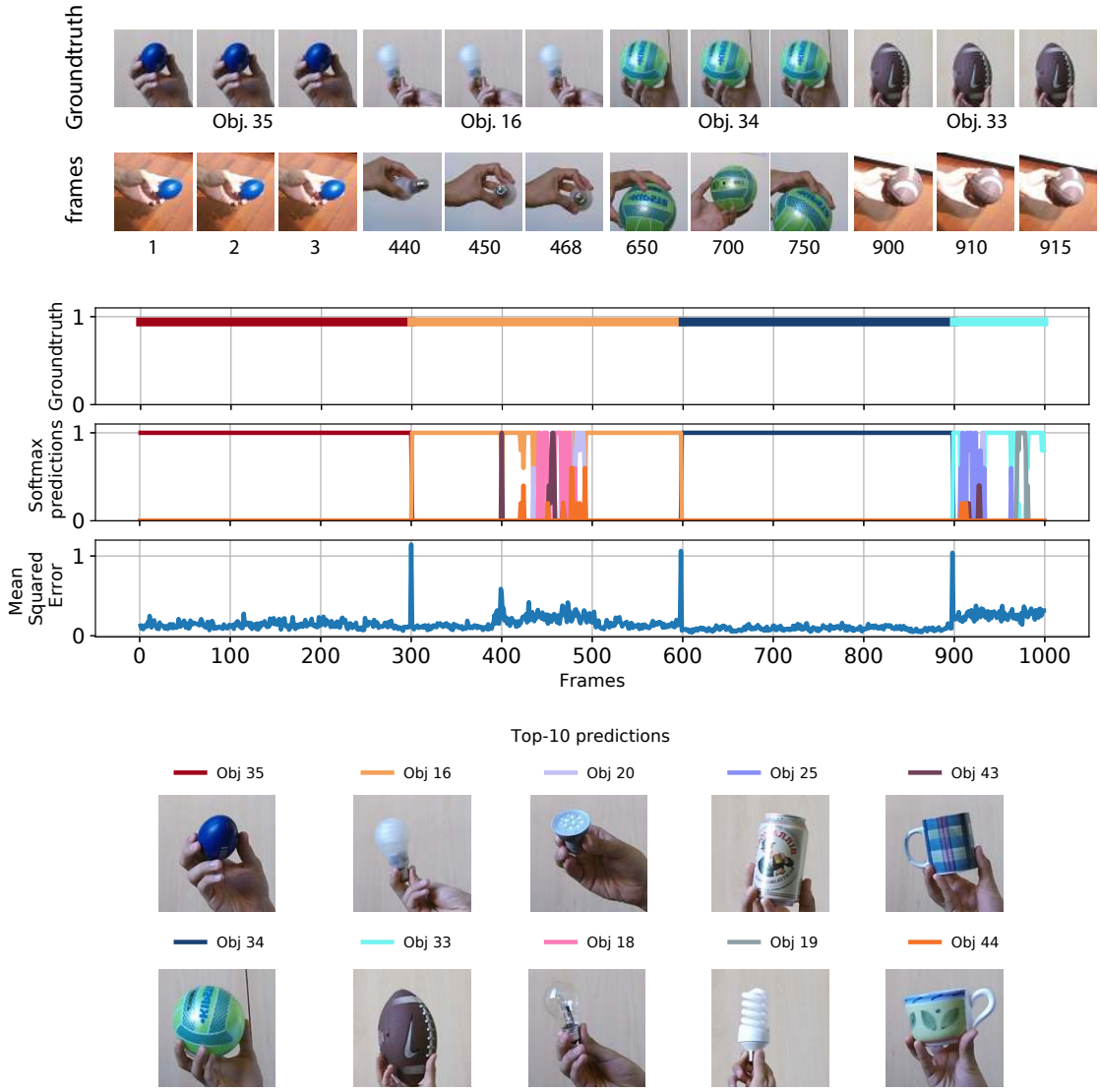


Figure 6.1: Euclidean distance between adjacent video frames, produced by our S-Triplet CNN, on the CORe50 dataset.

learning and classification losses, allowing the learning of novel objects without the need for retraining the model. While the S-Triplet model generates discriminative embeddings and achieves *state-of-the-art* recognition performance, it exhibits the same *sudden* changes between adjacent video frames as studied in Chapter 3.

We show an illustration of this behavior in Figure 6.1, where we compute the Euclidean distance of embeddings between adjacent frames, using the testing sequence in the CORe50 dataset. The features are produced by our S-Triplet model, using the methodology described in the previous chapter. We notice how there are abrupt changes

every 300 frames, which are explained by the change in the object and the background. However, there are also abrupt changes caused by ambiguous viewpoints, as illustrated in the frames 450-500, where the user rotates the bulb showing only the bottom part of it, which is identical in the other bulbs. Other examples of abrupt changes are the ones occurring in frame 900 onwards, where an American football is manipulated, and lateral views depict the football as a circular object, leading to being confused by top viewpoints of mugs and cans.

Discriminative features and robustness had been linked before in (Rozsa et al., 2017), where the authors highlight a capability in deeper models to learn representations that make features more separable and therefore, with higher accuracy and robustness. However, as we show in Figure 6.1, even models that are trained to be discriminative does not produce temporally consistent embeddings. Therefore, in this chapter, we aim to increase the robustness of the CNN by decreasing the distance between the embeddings of neighboring video frames, using a temporal filter that considers the distance from a previous video frame.

We propose the Centroids Triplet Network (CTN) for generating discriminative features by minimizing the distance between embeddings from available video frames, but also minimizing the distance respect to their *instance* centroid (a.k.a. *prototype*). Additionally, we speed up the inference stage by comparing a query embeddings against the centroids of each object manifold, as opposed to the nearest neighbors search in the embeddings space. We aim for a scalable real-time recognition system that can process *in-situ* data, as we argue that is a more straightforward approach of data collection compared to using complex data-acquisition setups, such as rotating tables, for obtaining pictures depicting them with ideal imaging conditions (*in-vitro*). Additionally, we aim for a robust recognition system that can deal with the commonly encountered changes in illumination, perspective, scale, backgrounds, and occlusions.

We evaluate our approach on datasets tailored for in-situ object recognition, which are characterized by presenting the objects with differing real-world training and testing conditions. We compare our approach against *state-of-the-art* methods for learning discriminative features and approaches that generate stable predictions in CNNs.

6.2 Literature Review

(Merler et al., 2007; Göhring et al., 2014) are pioneer works that address the problem of *in-situ* recognition of objects, understanding this as objects depicted within their natural

or common environments, as opposed to *in-vitro* pictures. (Merler et al., 2007) focuses on achieving domain generalization from *in-vitro* to *in-situ* data and (Göhring et al., 2014) combines features from *in-vitro* datasets with features extracted *in-situ* in order to build a large-scale real-time recognition system. As limitations, (Merler et al., 2007) requires *in-vitro* examples to extract clean descriptors and concluding that collecting *in-situ* data for training would be an impractical practice. On the other hand, (Göhring et al., 2014) requires that the class of the desired object to be learned is present in the ImageNet dataset to build a robust classifier.

More recently, the Amazon Robotics Challenge (ARC) 2017 presented the new requirement of learning novel objects efficiently by providing a set of unseen objects two hours before the competition. Since Amazon provided *in-vitro* images from such objects, winning teams (Zeng et al., 2018; Milan et al., 2017) proposed metric learning techniques to achieve domain adaptation between the images captured by the robot *in-situ* and the provided *in-vitro* images.

While (Zeng et al., 2018; Milan et al., 2017) were design for domain adaptation between *in-vitro* and *in-situ*, they empirically demonstrated that a CNN can be used to learn new objects without having to retrain the model by performing the nearest neighbors search in the features space, where data points from the same object are close to each other and separated otherwise. (Lagunes-Fortiz et al., 2018) builds on the same idea of utilizing a discriminative CNN and simplifies the model into a single branch that uses a combination of softmax and triplet loss for achieving *state-of-the-art* recognition performance for learning objects *on-the-fly*. We follow this research direction of utilizing discriminative networks for learning new objects efficiently, and therefore, we focus this literature review on supervised approaches that learn discriminative and robust features.

Regardless of the architectural design, it is now broadly studied that the commonly used combination of cross-entropy loss and the softmax function in the last fully connected layer, a.k.a. *softmax loss*, does not explicitly optimize the feature embedding to enforce higher similarity for intra-class samples and diversity for inter-class samples (Zheng et al., 2016; Liu et al., 2016b; Wen et al., 2016; Wang et al., 2017a; Lagunes-Fortiz et al., 2018; Wang et al., 2018; Deng et al., 2019; Eilertsen et al., 2019).

In this regard, the main approach for learning discriminable features consists of combining the softmax loss with regularizers that enforce the intra-class clustering and inter-class separation. These regularizes can be divided into Euclidean regularizers and angular-margin loss functions.

The *state-of-the-art* of Euclidean regularizers is center loss (Wen et al., 2016), where the CNN learns centers and the clustering of data points around those ones; triplet center loss (He et al., 2018), originally proposed for 3D object retrieval, proposes the incorporation of the triplet loss with center loss in order to enforce inter-class separability of clusters; Similarly, (Lagunes-Fortiz et al., 2018) combines the triplet loss with the softmax loss where the features for each loss are separated into two different heads, in order to improve the performance for recognizing novel objects. In all these approaches, an hyperparameter is used to balance the supervision signals.

On the other hand, CosFace (Wang et al., 2018), SphereFace (Wang et al., 2017a) and ArcFace (Deng et al., 2019) posits as the *state-of-the-art* angular-margin approaches and build on the findings from Large-margin Softmax (L-Softmax) (Liu et al., 2016b), which proposes a margin in the cosine product between the weights w and features x in the fully connected layer used for classification. As their name suggest, these approaches come from the facial recognition community and consist of angular constraints applied into the cosine version of the softmax loss. Additionally, for performing person identification, the authors utilize cosine similarity to compare a query feature, against the features in the database.

Producing stable predictions is also the focus of this work. Related to this regard, Stability Training (Zheng et al., 2016) and Single-Frame Regularization (Eilertsen et al., 2019) are the closest approaches to our goal. In Stability Training, the authors propose reducing the dissimilarity in the embeddings between an image X and a variant of it with a small perturbation $T(X) = X + \Delta X$, where the perturbation ΔX is described as per-pixel independent normal distributed noise $\Delta X \sim N(0, \sum)$, with $\sum = \sigma^2 I$. In (Eilertsen et al., 2019), where the goal is to achieve consistency for image-to-image translation, the authors proposes reducing the Euclidean between embeddings produced by an image X, where an affine transformation T has been applied before and after the translation: $\mathscr{L}_{trans-inv} = \|f(T(X)) - T(f(X))\|_2$.

As explored in Chapter 3, and studied by (Eilertsen et al., 2019), methods for enforcing temporal consistency in image processing are mostly based on estimating dense motion, optical flow, or using recurrent neural networks. While these approaches have shown usefulness, they all suffer from one or more of the following problems: 1) Training a CNN and a RNN are commonly separated training stages since one requires shuffled examples and the other sequential ones. 2) high complexity and application-specific architectural modifications, 3) a significantly increase in computational complexity for training and inference, 4) failure in situations where motion estimation is difficult, such as image

regions with occlusion or lack of texture. Since these limitations make the approaches above unsuitable for *in-situ* learning, we aim to achieve robustness using an external temporal filter that uses the embeddings produced by the CNN.

As detailed in section 6.4, our method utilizes instance centroids (a.k.a. *prototypes*) for speeding up the inference time and reduce the memory requirements during deployment. Prototypical Networks (Snell et al., 2017) and SimNet (Pinheiro, 2018) are related works that also utilizes centroids as embeddings to describe an object class. However, our work differs from Prototypical Networks by utilizing all training examples to compute the centroids, as opposed to a number of supporting examples per class (either one for one-shot learning, or five, for five-shot learning), additionally, we compute the centroids every number of epochs N, as opposed to calculate centroids at every training mini-batch. Similarly, our work differs from SimNet by comparing each data point against its corresponding centroid and a centroid from a negative example, and not all the possible centroids, which can be infeasible to compute in a dataset with hundreds of objects. Additionally, both methods utilize a trainable module to estimate the closest centroid given a query data point, while we propose an equivalent to the nearest neighbors search algorithm, which can be applied to new objects *on-the-fly*.

6.3 In-Situ Household Dataset

We collected a dataset consisting of 20 objects to test our approach in a genuinely *in-situ* scenario. The training and testing environments correspond to a kitchen, living room and bathroom, where we captured images of everyday household objects, without hands presence, and with varying imaging conditions between training and testing data.

6.3.1 Dataset Collection

We selected 20 objects, as the initial stage of our dataset, intending to add more places and objects in the future. We chose a kitchen, a living room and a bathroom. Within the kitchen we selected two cups, a milk carton, a knife, a honey bottle, a can opener, a cereal bag, an oil bottle, a soy sauce bottle, and a toaster; within the living room, we selected a pair of TV remotes, a duct tape, a potted plant and a toy; in the bathroom we chose a toothpaste, a toilet brush, a cleaning bottle, and a pair of shampoo bottles.

For collecting the images, within each environment, we choose a place with a clean background, to avoid learning features in the background or other objects, but at the

same time avoiding the use of complex data acquisition devices. To do so, we used the top part of a bookshelf, the kitchen counter, and a sofa.

We utilize an Android 10 - Pixel 2 phone for taking the pictures with the OpenCamera App to take PNG images with a resolution of $1,200 \times 1,200$ pixels, we then reduced the resolution to 480x480 since most *state-of-the-art* CNNs uses a resolution of 224×244. In Figure 6.2, we show an example of each object, as well as the training and testing conditions.

Figure 6.2: Objects collected for the In-Situ Household dataset.

6.4 Proposed Method

Our work builds on the findings from Chapter 5 for learning object *on-the-fly*. However, to make the scalable real-time recognition system that we are after we propose the following contributions. First, we replace the costly k nearest neighbors search as in (Zeng et al., 2018; Lagunes-Fortiz et al., 2018) by the normalized nearest centroids search in the features space. Secondly, we propose an additional distance constraint that enforces discriminability between embeddings and their instance centroids. Third, to exploit the temporal consistency in the video frames, we propose a temporal filter that takes into account predictions from previous frames as well as the distance between them. We now explain each of these contributions:

6.4.1 Learning Discriminative Features On-the-fly

As studied in (Zheng et al., 2016), unstable learners can classify neighboring video frames inconsistently due to visual perturbations such as noise. In this chapter, we aim to empirically demonstrate that in these unstable classifiers, there is a direct relation between inconsistent predictions and dissimilar embeddings. Furthermore, by enforcing the similarity between neighboring video frame embeddings, it is possible to make CNNs more robust against such nuisances.

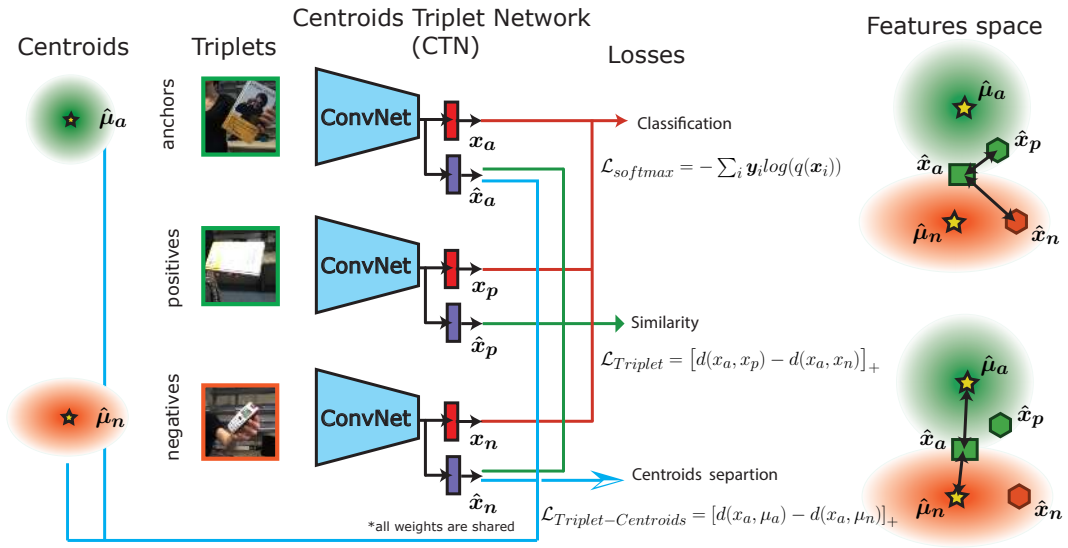
Since we are performing the nearest centroids search during deployment, we first propose a regularization term for minimizing the Euclidean distance between an anchor embedding and its corresponding centroid ($d(\hat{x}_a, \mu_a)$) and the Euclidean distance respect to a centroid taken from another objects $d(\hat{x}_a, \mu_n)$, as show in the following equation:

$$\mathscr{L}_{Triplet-Centroids} = \left[d(\hat{x}_a, \mu_a) - d(\hat{x}_a, \mu_n) \right]_+ \tag{6.1}$$

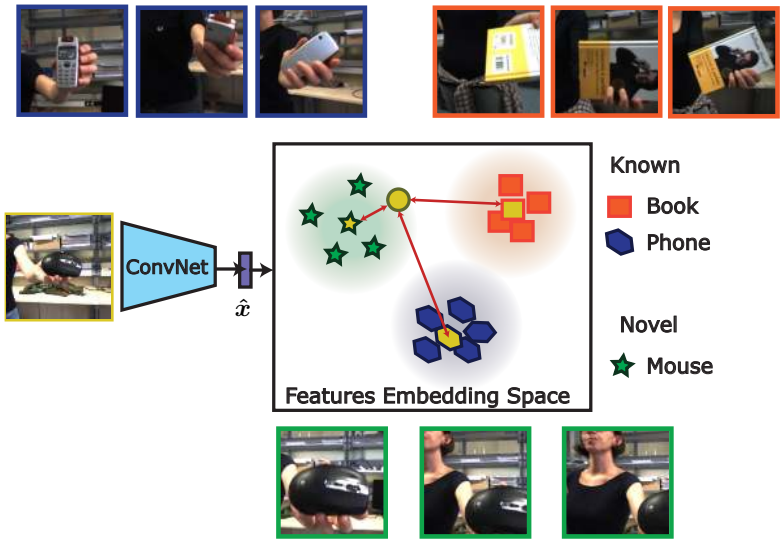
This formulation is fundamentally different from the Stability Training (Zheng et al., 2016) and Single-Frame Regularization (Eilertsen et al., 2019) approaches, where artificial perturbations are obtained by injecting Gaussian noise X', as well as applying affine transformations T into a given image X, and then minimizing the euclidean distance between the embeddings.

The final loss function that we propose to learn discriminative features is the following:

$$\mathscr{L}_{CTN} = \mathscr{L}_{Classification} + \alpha \cdot \mathscr{L}_{Similarity} + \beta \cdot \mathscr{L}_{Triplet-Centroids} \tag{6.2}$$



(a) Training



(b) Inference

Figure 6.3: Proposed object recognition framework for *in-situ* learning. Our Centroid Triplet Network (a). Classification is performed by the nearest centroids search in the features space (b).

$$\mathscr{L}_{Classification} = \mathscr{L}_{Softmax} = -\sum y log(q(x)) \tag{6.3}$$

$$\mathscr{L}_{Similarity} = \mathscr{L}_{Triplet} = \left[d(\hat{x}_a, \hat{x}_p) - d(\hat{x}_a, \hat{x}_n)\right]_+ \tag{6.4}$$

$$\mathscr{L}_{Triplet-Centroids} = \left[d(\hat{x}_a, \mu_a) - d(x_a, \mu_n)\right]_+ \tag{6.5}$$

To train our approach efficiently, we compute all the centroids every number of epochs n, as oppose to every mini-batch iteration. Furthermore, we utilize only a negative centroid per example to compare, as oppose to compare each embedding against all possible centroids. We show this in Figure 6.3.

6.4.2 Accelerating Inference Time

During deployment, we use the embeddings produced by the dense layer in purple from Figure 6.3. We first compute the centroids $\hat{\mu}_l$ by utilizing the labeled examples $(\hat{x}_1, y_1), (\hat{x}_2, y_2), ...(\hat{x}_n, y_n)$. We them sum all the embeddings belonging to the same object i, and then normalize the resulting vector as indicated in the equation:

$$\hat{\mu}_l = \frac{1}{\|\sum_i (\hat{x}_i)\|} \sum_i (\hat{x}_i) \tag{6.6}$$

To estimate the identity $\hat{y}$ from a given embedding $\hat{x}$, of a query image, we selected the identity of the closest center $\hat{\mu}$ found:

$$\hat{y} = argmin_{l \in Y} \|\hat{\mu}_l - \hat{x}\| \tag{6.7}$$

6.4.3 Temporally-Consistent Embeddings

To exploit the temporal information during deployment, we propose a temporal filter that enforces temporal consistency on embeddings from neighboring video frames. To do so, we propose a weighted average between the current and previous embeddings. The temporal filter is applied if the Euclidean distance d between the current and previous

video frame embeddings is lower than a threshold δ. A sudden *large* Euclidean distance d, defined by the hyperparameter δ, suggests a different object on the scene, and therefore, we give priority to the current representation. We describe the temporal filter in the following equation:

$$\hat{y}_t = \begin{cases} argmin_{l \in Y} \|\hat{\mu}_l - \hat{x}_t\|, & \text{if } d(x_t, x_{t-1}) \geq \delta \\ (1-\gamma) \cdot argmin_{l \in Y} \|\hat{\mu}_l - \hat{x}_t\| + \gamma \cdot \hat{y}_{t-1}, & \text{otherwise} \end{cases} \tag{6.8}$$

Where the values γ and δ are to be found empirically.

6.5 Experimental Results

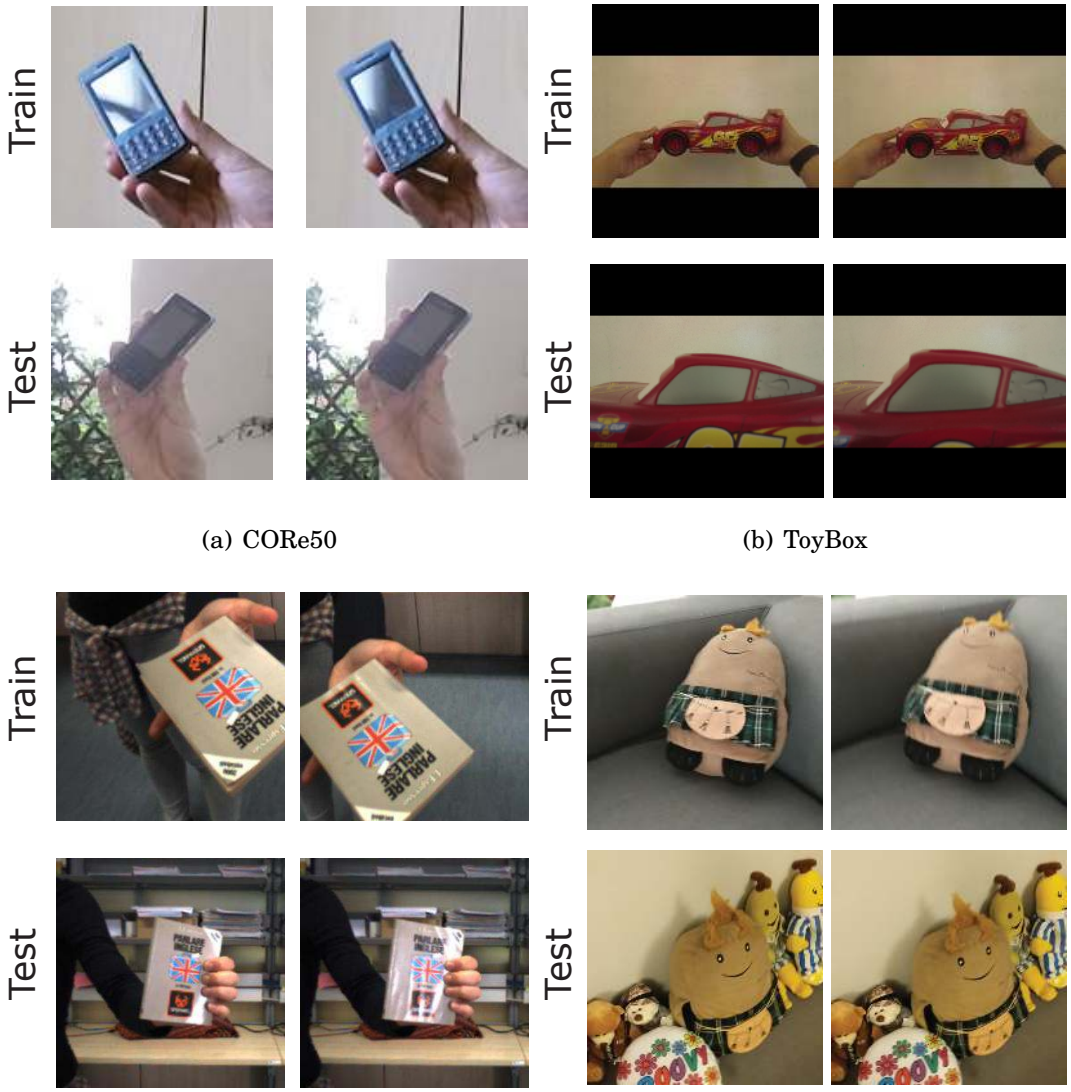
6.5.1 Datasets

We selected four datasets that depict objects from an egocentric view, as would be seen from a robot's perspective for most mobile robots and manipulators. We show an example of each dataset in Figure 6.4 and the remaining objects in Appendix A. Apart from the iCub dataset, where an actual robot took images, we selected datasets that present recognition scenarios that emulate the *in-situ* learning that we are after:

CORe50 (Lomonaco and Maltoni, 2017): Originally proposed for continuous learning, this dataset shows 50 objects across eleven environments and allows us to test the generalization capabilities not only to unseen object poses but also new environments. We utilize the standard testing set composed by scene 3, 7 and 11, and only the scene 1 for training the model. This results in 14,989 images training and 44,972 images for testing.

ToyBox (Wang et al., 2017c): It is composed of 360 toys manipulated by a person. Toybox allowed us to evaluate how well the model scales, by learning 120 novel objects *on-the-fly*. We utilize the hodgepodge videos for training and the translations and rotations across x,y,z-axis for testing. The testing set also depicts new conditions such as changes in scale, partial views, and occlusions. We use 24,667 images for training and 69,777 images for testing.

iCub transformations (Maiettini et al., 2017): This dataset contains 200 household objects shown by a demonstrator to an iCub robot. We selected the mixed manipulations set, taken with the left camera for training the model and mixed manipulations set but the following day for testing. The testing set depicts additional backgrounds, viewpoints, and scale. This results in 65,761 images for training and 34,312 images for testing.



(a) CORe50 (b) ToyBox

(c) iCub (d) Household in-situ

Figure 6.4: Object Recognition Datasets

In-situ household: This is the dataset proposed in this chapter, consisting of 20 objects with deferring training and testing conditions, without hand presence but depicting a variety of viewpoints, scales, clutter, occlusions, and illumination conditions. We use the total 5,957 images for training and 6,125 images for testing.

6.5.2 Baselines

To evaluate the effectiveness of our method, we selected *state-of-the-art* approaches for learning discriminative features, as it is relevant when learning objects *on-the-fly*, as well as approaches that aim for robust and temporally consistent predictions without the use of recurrent connections. All the backbone CNN in the following models consist of ResNet-50, with an additional Dense Layer with a dimension of 512 elements which is used as an embedding. We initialized all the models with weights learned from ImageNet, and we use stochastic gradient descent as optimizer with a learning rate of $l_r = 1 \times 10^{-3}$ and momentum $\mu = 0.9$. We selected the following hyperparameters for each baseline, each of them are explained in their corresponding citation:

Stability Training (Zheng et al., 2016): We utilize $\alpha = 0.01$, and for generating random noise, we use a standard deviation of $\sigma = 0.04$.

Invariance Regularization (Eilertsen et al., 2019): We utilize $\alpha = 0.95$ and the affine transformations described in (Eilertsen et al., 2019).

Center Loss (Wen et al., 2016): We selected the hyperparameters $\lambda = 0.1$ and $\alpha = 0.005$ in the loss function (Wen et al., 2016) and utilized the PyTorch implementation from (Zhou, 2018).

S-Triplet (Lagunes-Fortiz et al., 2018): We selected the hyperparameters $\lambda = 0.0001$ in the loss function (Wen et al., 2016) and utilized the PyTorch implementation from (Lagunes-Fortiz, 2019a).

Angular-margin approaches: **CosFace** (Wang et al., 2018), **SphereFace** (Wang et al., 2017a) and **ArcFace** (Deng et al., 2019) posit as the *state-of-the-art* angular-margin approaches for learning discriminative features. We utilize the generalized loss function presented in Equation 6.9, where the parameters m_1, m_2, and m_3 represents the constraint proposed in CosFace, SphereFace and ArcFace, respectively. As hyperparameters, we selected $m_1 = 0.35$, $m_2 = 0.5$, $m_3 = 4$, and $s = 30$. We utilize each margin at the time, leaving the remaining two equals to zero. As in their original implementations, we utilize cosine distance to measure the similarity between embeddings.

$$\mathscr{L}_{softmax} = -\log\left(\frac{e^{W_{y_i}x_i}}{\sum_j e^{W_{j_i}x_i}}\right) = -\frac{1}{N}\sum_{i=1}^{N}\log\left(\frac{e^{s(\cos(m_1\theta_{y,i}+m_2)-m_3)}}{e^{s(\cos(m_1\theta_{y,i}+m_2)-m_3)}\sum_{j=1,j\neq y_i} e^{s\cos(\theta_j)}}\right) \tag{6.9}$$

Softmax Loss: We utilize the most commonly used loss function for classification, consisting of a combination of cross-entropy loss with a softmax operation.

6.5.3 Implementation Details

We initialize the backbone CNN from a pre-trained model with Imagenet. Thus, our model works with RGB images with size 224×224. For training our model, we use mini-batches of 64 images, we use stochastic gradient descent with a learning rate of $l_r = 1 \times 10^{-3}$, momentum $\mu = 0.9$ and weight decay regularization of $wd = 1 \times 10^{-4}$. Code with training scripts is available at (Lagunes-Fortiz, 2019a).

We trained every model three times using the same train/testing split. We show the mean and standard error accuracy, in order to evaluate the stability of each model.

6.5.4 Recognizing Known and Novel Objects On-the-fly

To get an understanding of the usefulness of our proposed model, we show in Tables 6.1 - 6.3 the recognition performance for three different situations: (a) Only using known objects, (b) only using novel object and (c), a general case where there is no assumption about the object to test, and embeddings of known and novel objects are used for estimating the identity of a query object.

Adding the centroids triplet loss on the Supervised Triplet was particularly beneficial for the ToyBox dataset, gaining 4.9%, 3.9% and 3.3% for the known, novel and combined cases respectively. For the Core50 dataset, it showed a slight but consistent increase of 0.1%, 1.4% and 0.7% for each corresponding known, novel and combined cases. Similarly, for the iCub dataset, it show a slight but consistent performance of 0.8%, 1.9% and 1.6% cases. Finally, using our proposed dataset, we achieved an increase of 2.5% and 8.9% for the known and combined cases but a decrease of 4.9% for the case of recognizing only novel objects.

Overall, for the case of known objects, our model achieved an average 2.0% higher performance than the closest baseline, the S-Triplet model. For the case of novel objects, it achieved an average increased of 0.6%, and it achieved an average 3.6% higher recognition performance for the general case.

Table 6.1: Recognizing known objects % Accuracy top-1 recognition

	Core50	ToyBox	iCub	in-situ household	Average
Cross-Entropy Loss	69.50 ± 0.95	71.73 ± 0.91	**93.57 ± 1.11**	64.95 ± 1.12	74.93 ± 1.02
Stability Training (Zheng et al., 2016)	71.73 ± 0.87	71.91 ± 0.85	90.53 ± 0.85	79.14 ± 1.15	78.33 ± 0.93
Invariance Regularization (Eilertsen et al., 2019)	74.86 ± 0.7	72.52 ± 0.98	92.81 ± 0.98	**82.05 ± 1.17**	80.56 ± 0.95
CosFace (Wang et al., 2018)	59.83 ± 1.14	**86.11 ± 1.31**	92.85 ± 1.02	66.46 ± 1.23	76.31 ± 1.18
SphereFace (Wang et al., 2017a)	66.58 ± 1.34	83.72 ± 0.94	91.64 ± 1.13	67.37 ± 1.01	77.32 ± 1.11
ArcFace (Deng et al., 2019)	63.30 ± 1.01	85.91 ± 1.32	92.28 ± 1.34	66.18 ± 1.13	76.91 ± 1.2
Center Loss (Wen et al., 2016)	57.58 ± 0.95	76.86 ± 1.11	89.19 ± 1.07	78.02 ± 1.13	75.41 ± 1.07
S-Triplet (Lagunes-Fortiz et al., 2018)	74.47 ± 1.15	80.26 ± 1.12	92.68 ± 1.11	77.39 ± 1.19	81.21 ± 1.14
CTN (ours)	**74.53 ± 1.06**	85.13 ± 1.03	93.42 ± 1.01	79.82 ± 1.16	83.23 ± 1.18

Table 6.2: Recognizing novel objects *on-the-fly* % Accuracy top-1 recognition

	Core50	ToyBox	iCub	in-situ household	Average
Cross-Entropy Loss	59.93 ± 1.13	69.73 ± 0.95	79.18 ± 1.03	72.98 ± 1.29	70.46 ± 1.01
Stability Training (Zheng et al., 2016)	60.13 ± 0.98	75.02 ± 0.72	79.09 ± 1.02	80.20 ± 1.25	73.61 ± 0.99
Invariance Regularization (Eilertsen et al., 2019)	56.92 ± 0.87	73.77 ± 0.91	80.37 ± 1.09	79.31 ± 1.15	72.60 ± 1.00
CosFace (Wang et al., 2018)	59.83 ± 1.24	**82.32 ± 1.41**	80.86 ± 1.27	**95.01 ± 1.13**	79.51 ± 1.26
SphereFace (Wang et al., 2017a)	62.53 ± 1.19	81.11 ± 1.34	79.09 ± 1.33	90.08 ± 1.11	78.20 ± 1.24
ArcFace (Deng et al., 2019)	61.93 ± 1.11	80.89 ± 1.28	77.40 ± 1.13	94.76 ± 1.02	78.75 ± 1.13
Center Loss (Wen et al., 2016)	57.58 ± 1.01	51.96 ± 1.09	49.87 ± 1.04	81.30 ± 1.11	60.18 ± 1.06
S-Triplet (Lagunes-Fortiz et al., 2018)	64.67 ± 1.15	77.30 ± 1.05	82.36 ± 1.11	90.40 ± 1.51	79.18 ± 1.21
CTN (ours)	**66.11 ± 1.09**	81.23 ± 1.05	**84.11 ± 1.64**	85.49 ± 1.21	**79.23 ± 1.18**

The temporal filter that uses the embeddings distances resulted consistently useful for increasing the performance for all the approaches, as we show in Table 6.4. For our model, there was an average increase of 7.33% higher performance for the general case of known and novel objects combined. With the model still making mistakes when there are ambiguous viewpoints from the very beginning of a testing sequence.

Table 6.3: Recognizing known and novel objects *on-the-fly* % Accuracy top-1 recognition

	Core50	ToyBox	iCub	in-situ household	Average
Cross-Entropy Loss	52.84 ± 1.14	62.73 ± 0.85	81.66 ± 1.02	50.14 ± 1.19	61.84 ± 1.05
Stability Training (Zheng et al., 2016)	53.73 ± 1.19	71.96 ± 0.98	79.85 ± 1.02	71.96 ± 1.13	69.38 ± 1.08
Invariance Regularization (Eilertsen et al., 2019)	52.62 ± 1.05	72.52 ± 0.93	82.05 ± 1.11	72.52 ± 1.15	69.93 ± 1.06
CosFace (Wang et al., 2018)	49.61 ± 1.02	80.53 ± 1.51	81.57 ± 1.14	66.05 ± 1.21	69.44 ± 1.22
SphereFace (Wang et al., 2017a)	53.06 ± 1.13	78.12 ± 1.24	80.59 ± 1.23	63.17 ± 1.17	68.74 ± 1.20
ArcFace (Deng et al., 2019)	50.71 ± 1.04	79.65 ± 1.21	79.96 ± 1.35	64.16 ± 1.01	68.62 ± 1.15
Center Loss (Wen et al., 2016)	44.92 ± 1.21	59.76 ± 1.08	63.30 ± 1.01	58.80 ± 1.12	56.69 ± 1.10
S-Triplet (Lagunes-Fortiz et al., 2018)	55.61 ± 1.05	78.26 ± 1.12	82.82 ± 1.02	64.37 ± 1.32	70.26 ± 1.13
CTN ours	**56.31 ± 1.14**	**81.55 ± 1.02**	**84.32 ± 1.16**	**73.25 ± 0.97**	**73.85 ± 1.09**

Table 6.4: Recognizing known and novel objects *on-the-fly* with Temporal Filtering % Accuracy top-1 recognition

	Core50 $\alpha = 0.95, \delta = 4$	ToyBox $\alpha = 0.95, \delta = 4$	iCub $\alpha = 0.95, \delta = 5$	in-situ household $\alpha = 0.98, \delta = 4$	Average
Cross-Entropy Loss	59.19 ± 1.14	64.37 ± 0.85	85.21 ± 1.02	55.14 ± 1.19	65.94 ± 1.05
Stability Training (Zheng et al., 2016)	66.52 ± 1.19	76.27 ± 0.98	84.31 ± 1.02	77.01 ± 1.13	76.03 ± 1.08
Invariance Regularization (Eilertsen et al., 2019)	66.81 ± 1.05	78.10 ± 0.93	86.05 ± 1.11	79.29 ± 1.15	77.56 ± 1.06
CosFace (Wang et al., 2018)	63.75 ± 1.02	86.87 ± 1.51	88.97 ± 1.14	73.21 ± 1.21	77.56 ± 1.22
SphereFace (Wang et al., 2017a)	66.08 ± 1.13	83.45 ± 1.24	85.31 ± 1.23	71.17 ± 1.17	76.50 ± 1.20
ArcFace (Deng et al., 2019)	62.33 ± 1.04	82.64 ± 1.21	86.22 ± 1.35	69.13 ± 1.01	75.08 ± 1.15
Center Loss (Wen et al., 2016)	61.47 ± 1.21	72.32 ± 1.08	68.30 ± 1.01	65.21 ± 1.12	66.82 ± 1.10
S-Triplet (Lagunes-Fortiz et al., 2018)	67.51 ± 1.05	83.12 ± 1.12	88.74 ± 1.02	77.98 ± 1.32	79.48 ± 1.13
CTN ours	**68.05 ± 1.14**	**87.01 ± 1.02**	**92.30 ± 1.16**	**81.25 ± 0.97**	**82.14 ± 1.09**

6.5.5 Scalability and Real-Time Inference

In Figure 6.5 we present the improvement in storage required and inference time between the commonly used k-nearest neighbors ($k = 5$, (Zeng et al., 2018; Lagunes-Fortiz et al., 2018)) search and the proposed normalized version of the nearest centroids algorithm. For each dataset, we considered the general case of testing *known* and *novel* objects (Table 6.3). We compare the recognition performance, the wall-clock time taken for evaluating all testing images and the storage required for saving all training embeddings and the instance centroids.

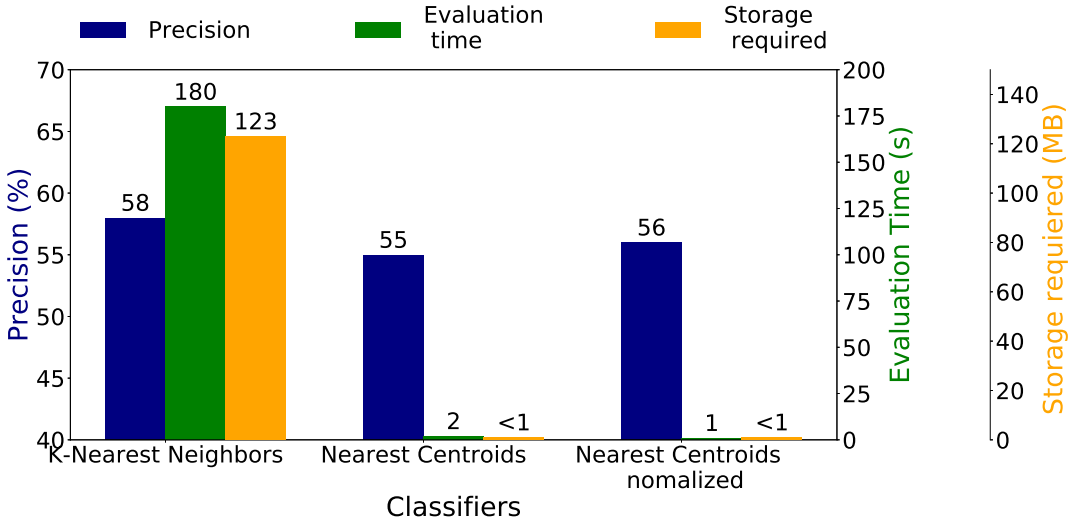
From Figure 6.5, we noticed a consistent decrease recognition performance of around 2% in each dataset of our normalized-nearest centroids algorithm against k-nearest neighbors. However, there is a considerably reduction in storage required and the inference time. For all the datasets, storing only the centroids represented less than 1% of storage required, compared to storing all training embeddings required in k-nearest neighbors. Related to the inference time, the nearest centers algorithm is two order of magnitude faster than k-nearest neighbors, making it an overall more suitable approach for a *scalable* and *real-time* object recognition system.

6.5.6 Hyperparameters Searching

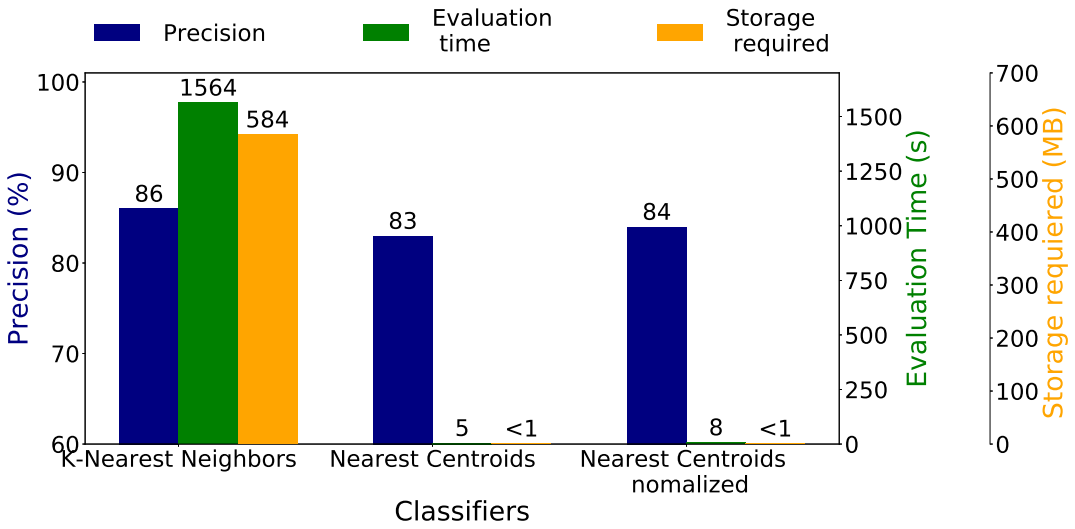
In order to find a suitable value for the controlling α and β hyperparameters in Equation 6.2, we perform a grid search, ranging values from 1×10^{-1} to 1×10^{-5} by decreasing an order of magnitude each step for both hyperparameters. We started by finding the best value for α by setting $\beta = 0$. With the best overall value of $\alpha = 1 \times 10^{-3}$, we found $\beta = 1 \times 10^{-2}$ to be the best overall value. In general, α and β with values grater than 1×10^{-2} causes a degradation in precision, and choosing values ranging 1×10^{-3} to 1×10^{-4} resulted in the best recognition performance. Therefore, we recommend choosing $\alpha = 1 \times 10^{-3}$ and $\beta = 1 \times 10^{-3}$ as starting points in other object recognition datasets.

6.5.7 Ablation Studies

As ablation studies, we first explore similarity metrics for comparing embeddings from neighboring video frames. We considered Manhattan distance, cosine similarity, and Euclidean distance. To evaluate the usefulness of the additional triplet-centroids loss, we compare our proposed model against the S-Triplet, a model trained with softmax loss and a model trained with the softmax and triplet-centroid loss. We found that using Euclidean distance resulted in the highest overall performance, as shown in Figure 6.7(a).



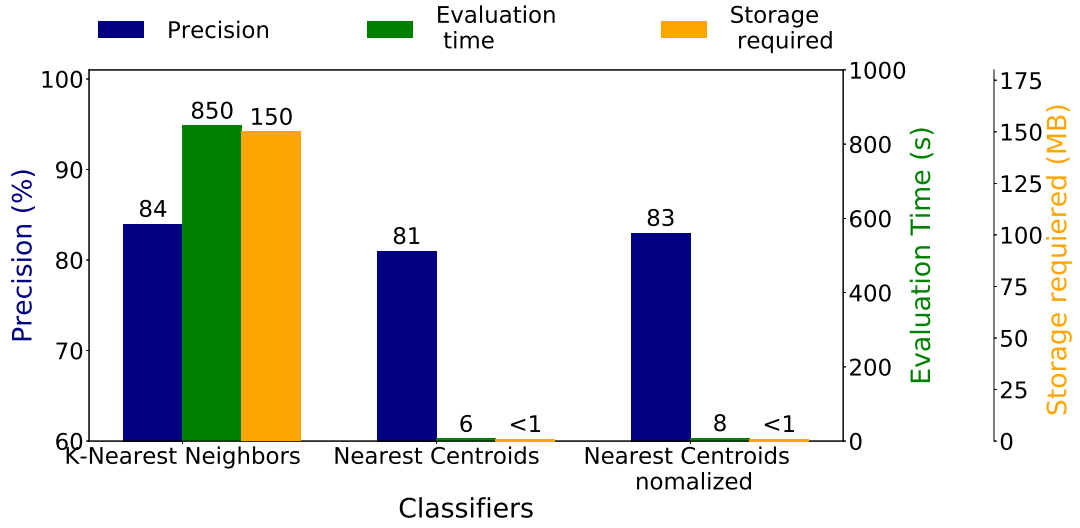
(a) Core50



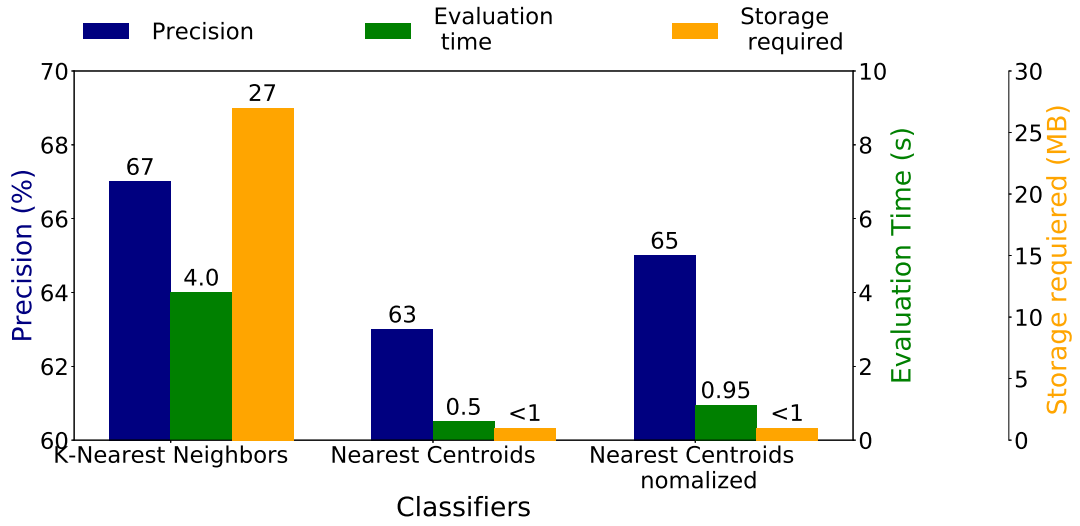
(b) ToyBox

Figure 6.5: Comparison between k-nearest neighbors and the proposed nearest centroids algorithm. We compare precision, evaluation time, and storage required.

We also explored different sizes for the embeddings, as we show in Figure 6.7(b). We observe that larger sizes sizes achieve an overall higher precision, with an approximate increase of 2% by doubling the embedding dimension. Noticeably, for the household dataset, there was a decrease in precision when using a dimension of 1024, indicating that additional regularization might be required.



(c) iCub

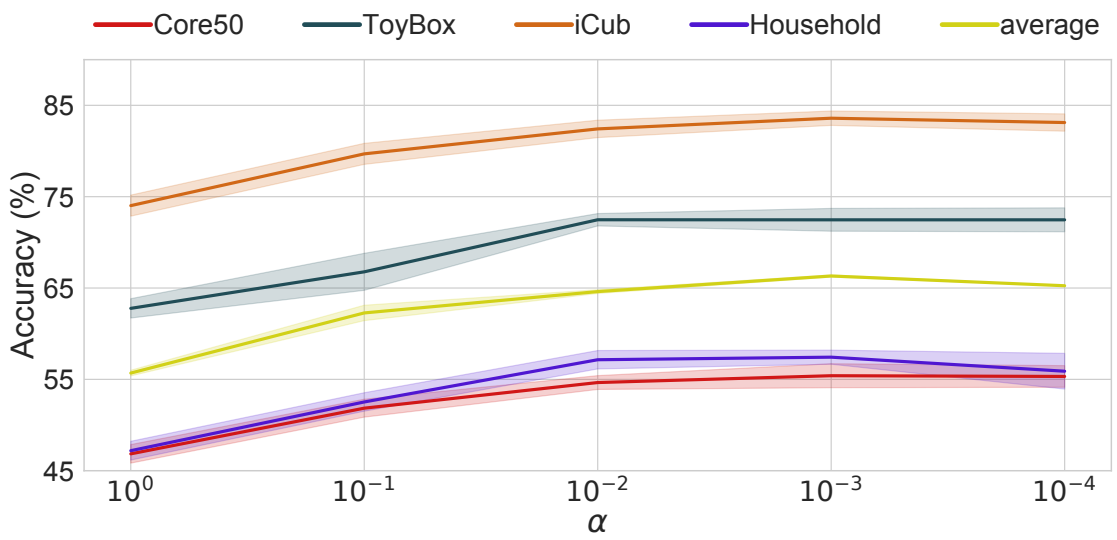


(d) Household

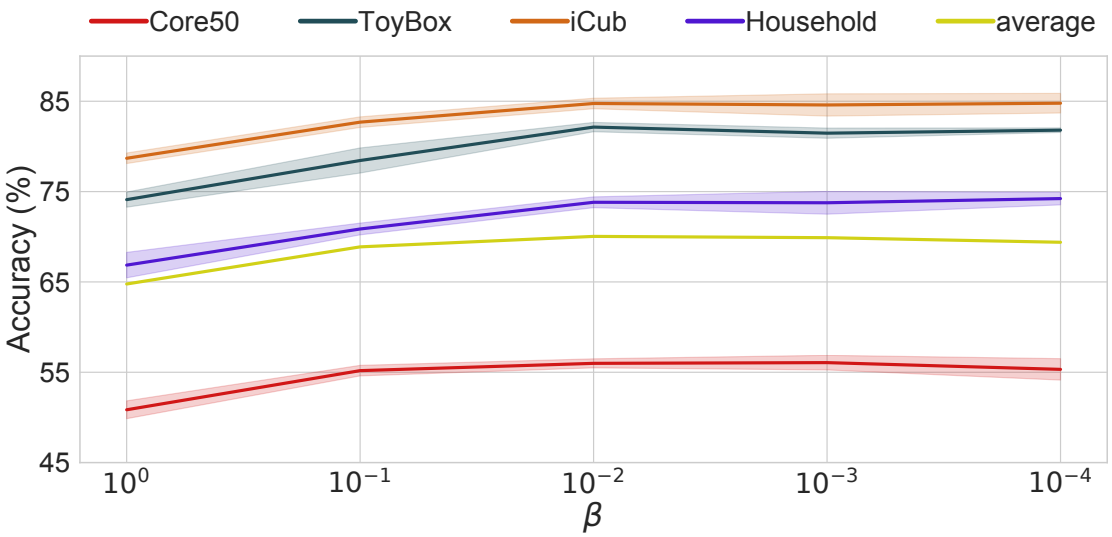
Figure 6.5: Comparison between k-nearest neighbors and the proposed nearest centroids algorithm. We compare precision, evaluation time, and storage required.

6.5.8 Discussion

As shown in Table 6.3, adding the triplet-centroids loss led to an average increase of 3.6% in precision. This increase in performance suggests that, with the in-situ datasets used, the features produced by the S-Triplet did not distributed as a *Gaussian hyper-sphere* around its centroid, as suggested by related work (Wang et al., 2017a, 2018; Deng et al., 2019), meaning that there were features closer to other instance centroid than



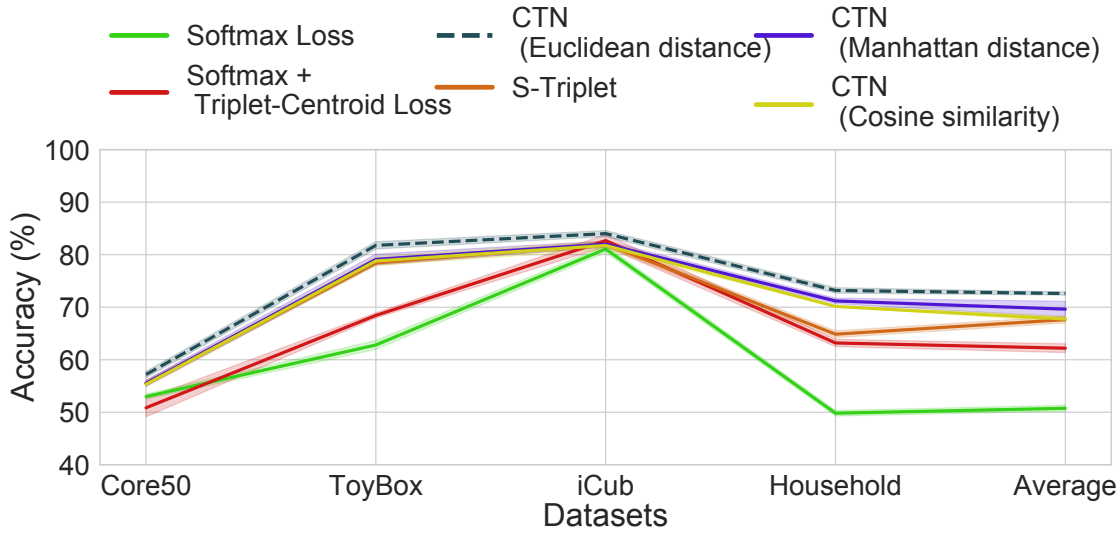
(a) Finding the best α.



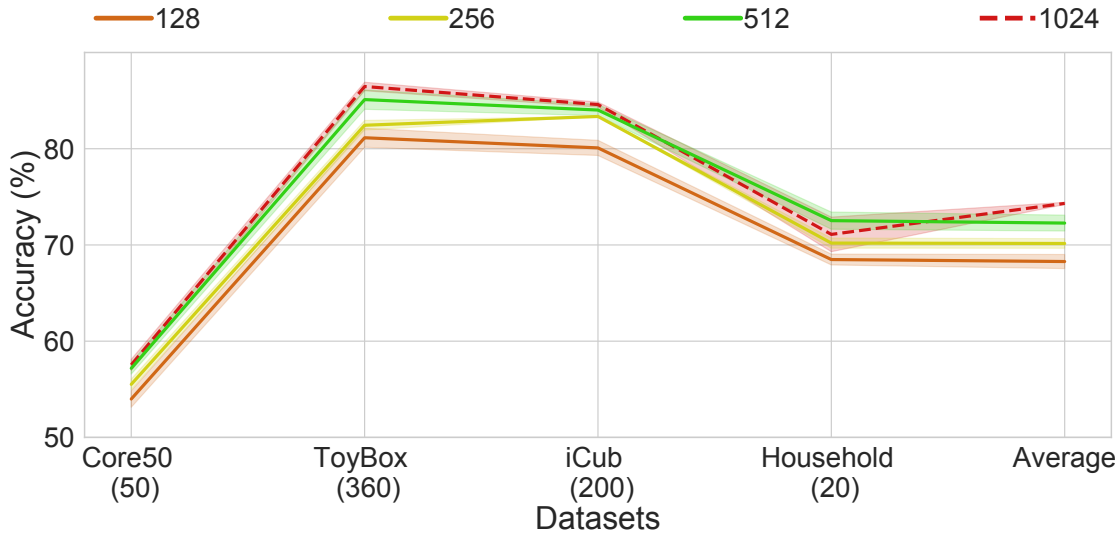
(b) Finding the best β.

Figure 6.6: Hyperparameters tuning.

its corresponding one, this was more notorious in the proposed household and iCub dataset, and less critical for the CORe50 dataset. In this regard, there is still more to be known about the properties of the manifolds generated in the *in-situ* learning scenario and we leave that as future research direction. Replacing the nearest neighbors search by the nearest centroids algorithm resulted in a highly beneficial approach for the onboard learning capabilities that we are after, since using this algorithm allowed a faster inference time by up-to two orders of magnitude, and a storage space of only 1.8



(a) Losses and distance metrics

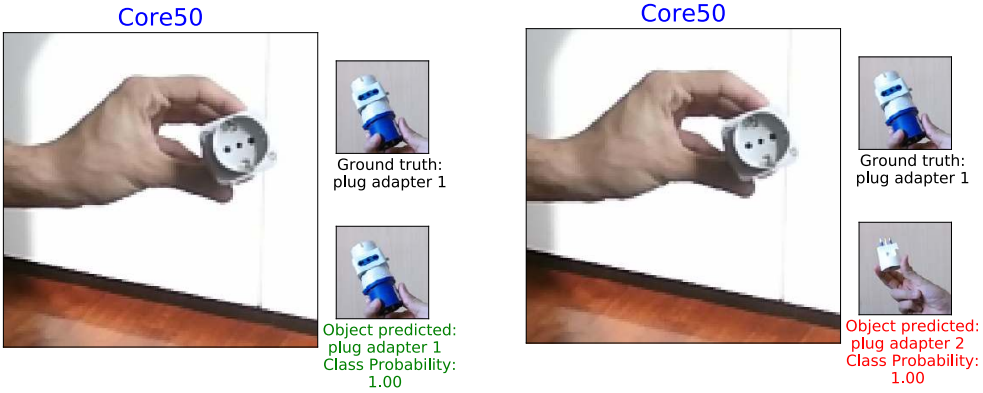


(b) Embeddings size

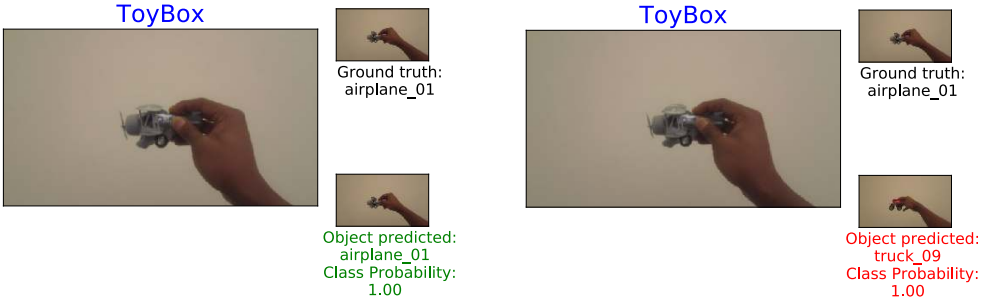
Figure 6.7: Ablation Studies. Image best viewed in the digital version.

KB of memory per object, using an embedding size of 512 elements with numpy 1.17 and using Python 3.7.

Respect to the proposed weighted average used as a temporal filter, while adding the embeddings separation helped to quickly react when new objects appear on the scene and gaining a further average increase of 7% in precision, there are cases where using a static threshold δ was not sufficient. Examples of this, are the misclassifications between known and novel objects, as shown in Figure 6.8. Where the Euclidean distance



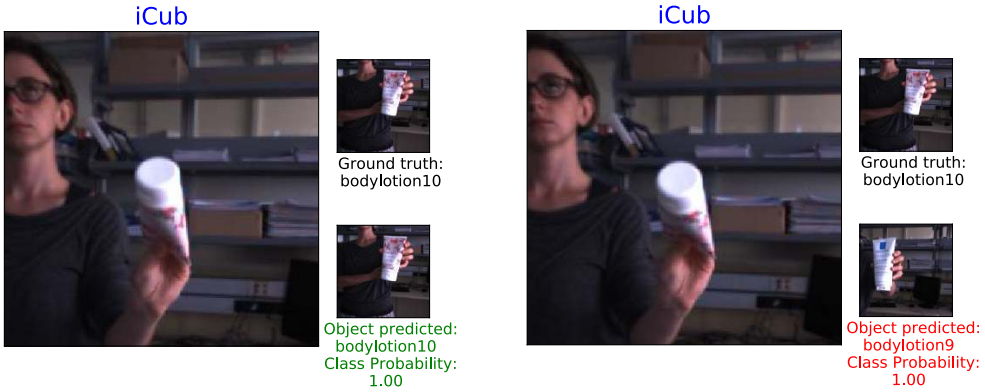
(a) Unseen adapter 1 misclassified as the known adapter 2.



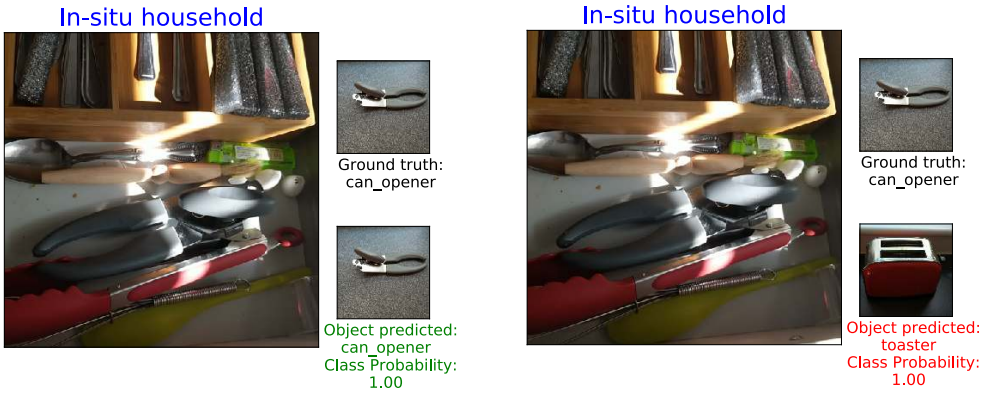
(b) Known airplane 1 misclassified as the unseen truck 9.

Figure 6.8: Misclassification cases ocurring between a known and a novel object. Despite applying the temporal filtering, the Euclidean distance between these neighboring embeddings is higher than the threshold δ, and thus allowing a misclassification. Image best view in the digital version.

between neighboring video frame exceeded the threshold δ, giving preference to the current estimation and ignoring the previous temporal information. This limitation calls for a temporal filter that can adjust their beliefs in a dynamic way, as new evidence is available about the object at hand. We leave the developments of this temporal filter as future work.



(c) Unseen bodycream 10 misclassified as the known bodycream 9.



(d) Unseen can opener misclassified as the known toaster.

Figure 6.8: Misclassification cases ocurring between a known and a novel object. Despite applying the temporal filtering, the Euclidean distance between these neighboring embeddings is higher than the threshold δ, and thus allowing a misclassification. Image best view in the digital version.

6.6 Summary

In this chapter, we focus on the *in-situ* learning of objects, consisting of learning generalizable embeddings from few training examples collected within the natural environment where the objects are normally used or placed. We built on the findings from Chapters 3 and 5, for proposing a model that not only generates discriminative features but will also consider the temporal consistency that must exist between nearby frames. Overall, for the case of known objects, our model achieves an average 2.01% higher performance than the closest baseline, the S-Triplet model. Furthermore, achieves higher precision compared to state-of-the-art approaches such as ArcFace and Stability Training ap-

proaches. For the case of novel objects, it achieves a slight increased of 0.61%, this is explained by the 5% decrease in performance in the household dataset, and it achieves an average of 3.21% higher recognition performance for the general case. Furthermore, by replacing the nearest neighbors search for the nearest centroids algorithm, we improve the inference time by up to two orders of magnitude and the data storage to less than a MB per dataset. Finally, we further increase the robustness against nuisances with an average increase of 7% for the known and novel cases, by proposing a temporal filter that uses the distance between neighbor embeddings to decide weather or not to apply the filtering condition.

CHAPTER 7

CONCLUSIONS

This chapter concludes the thesis by discussing the advantages and limitations of our proposed models for *in-situ* object recognition, as well as proposing possible research directions.

7.1 Contributions

In this thesis, we consider the problem of *in-situ* learning, consisting of a learner able to recognize objects from a few examples. As a core aspect, the training examples are collected within the environment where such objects are usually used, as opposed to using complex image capturing set-ups or synthetic images. Furthermore, we aim for a learner that can learn additional instances onboard, that can generalize under unseen environments, that is scalable and can operate in real-time conditions.

We empirically study the performance and limitations of *state-of-the-art* CNNs for multiview object recognition and identify the following flaws that prevent the use of off-the-shelf CNNs for *in-situ* learning:

1. Inconsistency in the predictions between neighboring video-frames.

2. Need for vastly training examples for achieving generalization.

3. Need for finetuning when learning additional objects.

Based on the findings of the research presented in this thesis, our contributions are the following:

- In Chapter 3, we propose the use of **temporal filters** on top of CNNs for addressing the lack of consistency in the predictions from neighboring video-frames. As the main advantage, the temporal filters are particularly useful for correcting misclassifications caused by sudden ambiguous viewpoints, such top-view in mugs, which are similar across all instances. On the other hand, temporal filters do not help week classifiers to improve performance, since they utilize the predictions from the CNNs to estimate predictions with temporal consistency. We find the cumulative moving average algorithm more suitable for *in-situ* learning since it does not require a training stage. In Chapter 3, we propose a generic version of it, consisting of a sum rule, and in Chapter 6, we propose a weighted sum that takes into account features separation between neighboring video-frames, which is beneficial for an agent that needs to react to changing objects in the scene. With our temporal filters, we achieve a precision of 99.13% for CORe50 and a mean average precision of 56.23% in T-LESS.

- In Chapter 4, we propose the use of a **multi-task CNN** that performs image classification and image generation to achieved domain generalization. Our approach results particularly useful for learning discriminative features, utilizing training examples drawn from a single domain, and generalize to an unseen one. For example, generalizes to *in-situ* images from in-vitro images. As a disadvantage, the multi-task CNN struggled to represent unseen objects and requires a fine-tuning stage for learning additional objects. We empirically evaluate our approach on eight object recognition datasets used for robotics and computer vision recognition tasks, and achieving an average 5.23% higher performance compared with the closest baseline and over 20% higher performance compared with a *state-of-the-art* adversarial network approach.

- In Chapter 3, we propose a discriminative CNN that classifies images and enforces discriminability on the learned features by utilizing a combined cross-entropy and triplet losses. Each of these tasks is perform by a separated dense layer. The layer trained with the triplet loss learned features that can generalize better to unseen objects, compared with the layer used for classification. Our discriminative model allows the learning of additional objects **on-the-fly**, by replacing the classification layer by the nearest neighbors search in the embeddings space. As a disadvantage,

the inference time and storage required can be compromised, as more training examples and instances are acquired, and used for the nearest neighbors search. Our supervised triplets network, had a comparable performance to the fine-tuned models across all datasets, it achieves an average 4% higher performance for known objects, 5% for novel ones and 7% higher for the general case of combined known and novel objects, compared with the closest baseline.

- In Chapter 6, we propose replacing the expensive nearest neighbors search for the **nearest centroids** algorithm to accelerate the inference time, and to reduce the storage requirements. Based on the nearest centroid algorithm, we propose a regularization term to enforce discriminability between features and their centroids. We propose utilizing the distance between embeddings from neighboring video-frames, to dynamically decide when to apply the sum-rule filter from Chapter 3. Finally, we propose a dataset for a systematic benchmarking of *in-situ* learning approaches. Overall, for the case of known objects, our model achieves an average 2.01% higher performance than the closest baseline, the S-Triplet network from Chapter 5. For the case of novel objects, it achieves a slight increase of 0.61%, an average of 3.21% higher recognition performance for the general case.

- Our final model for *in-situ* learning consists of a CNN that produces discriminative features by using a supervised triplet and triplet-centroids losses, that uses the nearest centroids algorithm, and the temporally-consistent embeddings filter at inference time, to achieve robustness against nuisances, keep scalability, and real-time performance. Using a device with a 6th generation Core i7 CPU, 16 GB RAM, a GTX-1060 GPU card, and using ResNet-50 as the backbone CNN, our model predicts the identity of an object in 1.15 ms, utilizes 1.08 KB of memory per object, it learns a new instance by computing and storing the centroid of a set of training images, which takes an estimate of 1.2 s when hundreds of training images are used, (such as the 300 images in the CORe50 and Household datasets).

7.2 Limitations

While our approaches achieve *state-of-the-art* recognition performance, the following aspects require further research:

Temporal filtering: The cumulative moving average, proposed in Chapter 3, results into a robust strategy for dealing with nuisances, but requires a careful selection of the

number of frames to be fused, and has a slow response for objects suddenly appearing on the scene. The weighted embeddings-filter from Chapter 6, has the benefit of quickly react to objects appearing on the scene but is less robust in cases where an object disappears, or is strongly occluded. Therefore, there is still room to improve the temporal filtering, with a model that can both react quickly to new objects on the scene, and that can handle strong nuisances.

Multi-task classification and image generation: The multi-task model from Chapter 4 utilizes a conventional autoencoder for generating images and would be interesting to explore if more modern approaches, such as generative adversarial networks, lead to an increase in recognition performance when they are used in combination with image classification.

Generating discriminant features *on-the-fly*: A challenging scenario is to learn similar instances *on-the-fly*, as we show in the failure cases in Chapter 6, as their features tends to be close to each other in the metric space, leading to misclassification. Although we enforce similar instances to have discriminative features during training, it was difficult to replicate such behavior for similar unseen instances during deployment. Therefore, while using a larger corpus of training data might help to achieve better generalization, it is still an open research question on how to generate discriminative features for unseen and similar instances.

7.3 Future Research Directions

We identified the following scenarios as further research directions.

Learning an unequivocal embeddings representation from a few data: A key challenge in multiview object recognition is that training a model without the full sphere of viewpoints can lead to a weak recognition performance, especially when the missing views are similar to the ones of other instances. While human intelligence offers the flexibility for efficiently updating its beliefs in the presence of new evidence as well as being able to extrapolate missing information, it is a challenge achieving this behavior in CNNs. While data augmentation in the features space like (Liu et al., 2018a; Yin et al., 2019) has shown usefulness when learning from canonical views, the recognition performance is still considerably lower compared to using the full sphere of views. Furthermore, there is more to be understood about the distribution of the features in the embedding space generated by the CNNs.

Dealing with ambiguous viewpoints: For similar instances, the proposed models

in this thesis tended to estimate with a high probability score the identity of the object, ignoring the presence of ambiguous viewpoints. Examples of ambiguous viewpoints are the images depicting the brass base only in the bulbs from the CORe50 dataset, where such metallic part is identical in all instances. Bayesian Deep Learning approaches such (Hall et al., 2018; Miller et al., 2019) aim to model the concept of aleatoric and epistemic uncertainty, which could be helpful for estimating the ambiguity in a given image. Furthermore, if an agent is aware of this ambiguity, a research question arises about what *action* should take to acquire more certain information.

In-situ object detection and segmentation: While object recognition can be used to solve a variety of challenges, it requires that object candidates occupy the majority of the image. Furthermore, it assumes there is only one object of interest in the image. A prior and correct object location is a strong assumption that limits our approach to be used into more challenging problems such as object manipulation or surveillance. Therefore, an immediate research direction is to learn to localize and segment objects *in-situ*. (Pinheiro et al., 2015) is a pioneer work for instance segmentation for unseen objects and (Wang et al., 2019) aims for an object detector that can work in any domain, giving some insights that both object detection and segmentation could be learned *in-situ*.

With systems capable of learning discriminative features of objects using a few examples, that can generalize in unseen environments and efficiently deal with nuisances. We will be able to move away from the conventional and static training or finetuning of CNNs for learning to recognize objects into more autonomous and human-like learning of objects.

Most importantly, we will be closer to developing autonomous systems that exhibit the generality and adaptability of human intelligence. That can efficiently learn and re-learn the concept of an object, making them more suitable to be used in our unstructured and uncertain world. It is my hope that the research question and framework presented in this thesis will lay the foundations for more contributions to the exciting field of *in-situ* learning.

APPENDIX A

A.1 Known and Novel Instances in Datasets

In this section we show the object in each dataset. The numbering is consistent with the confusion matrices from Chapter 3 and for Chapters 4, 5, and 6, we denote the objects belonging to the *known* set with blue and to the *novel* set with orange.

A.1.1 T-LESS

1 **Obj 1** 2 **Obj 2** 3 **Obj 3** 4 **Obj 4** 5 **Obj 5**

Figure A.1: TLESS

Figure A.1: TLESS

A.1.2 CORe50

1 Connector 1 2 Connector 2 3 Connector 3 4 Connector 4 5 Connector 5

6 Phone 1 7 Phone 2 8 Phone 3 9 Phone 4 10 Phone 5

11 Scissors 1 12 Scissors 2 13 Scissors 3 14 Scissors 4 15 Scissors 5

16 Bulb 1 17 Bulb 2 18 Bulb 3 19 Bulb 4 20 Bulb 5

21 Can 1 22 Can 2 23 Can 3 24 Can 4 25 Can 5

Figure A.2: Objects in CORe50.

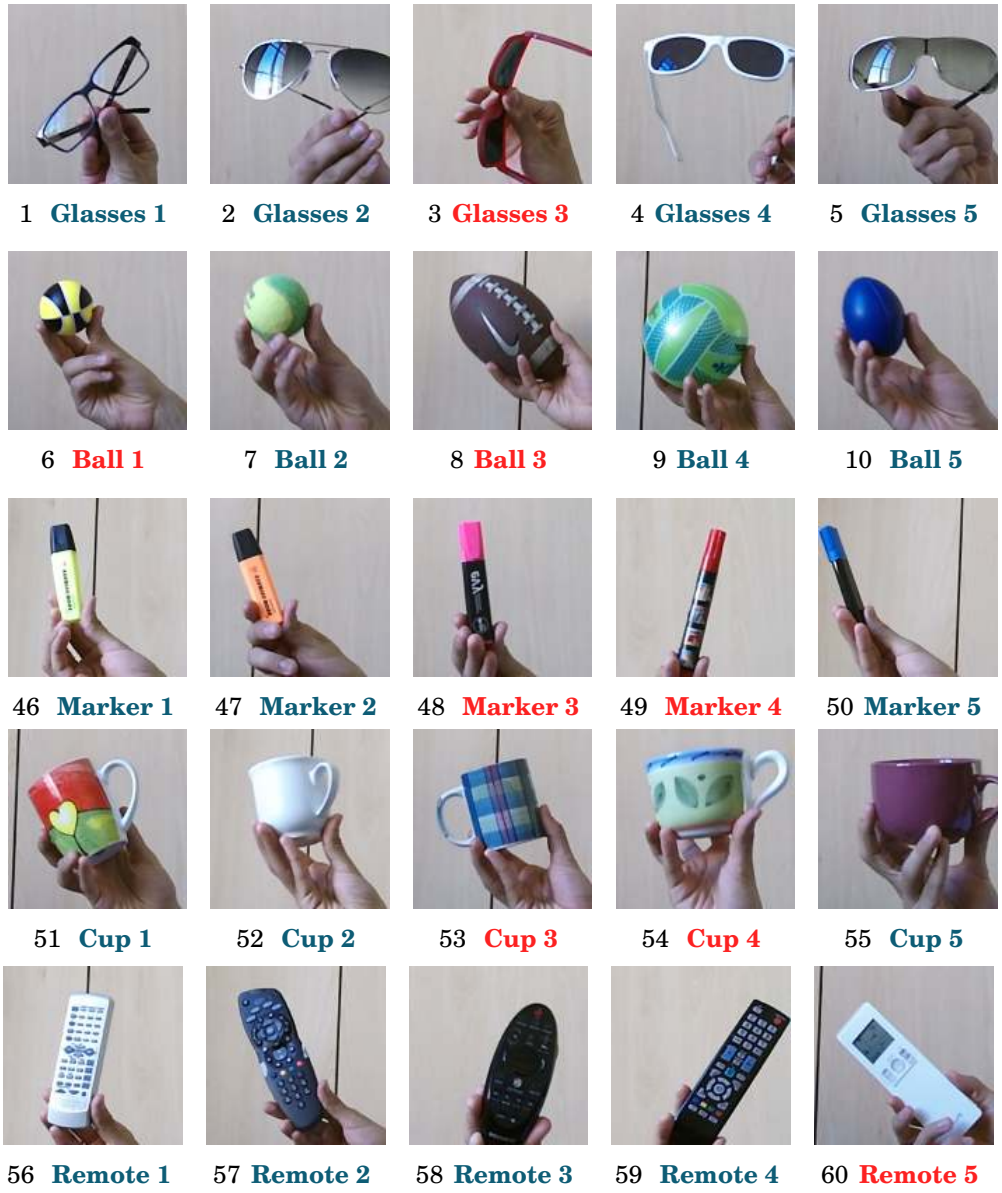


Figure A.2: Objects in CORe50. (continuation)

A.1.3 ToyBox

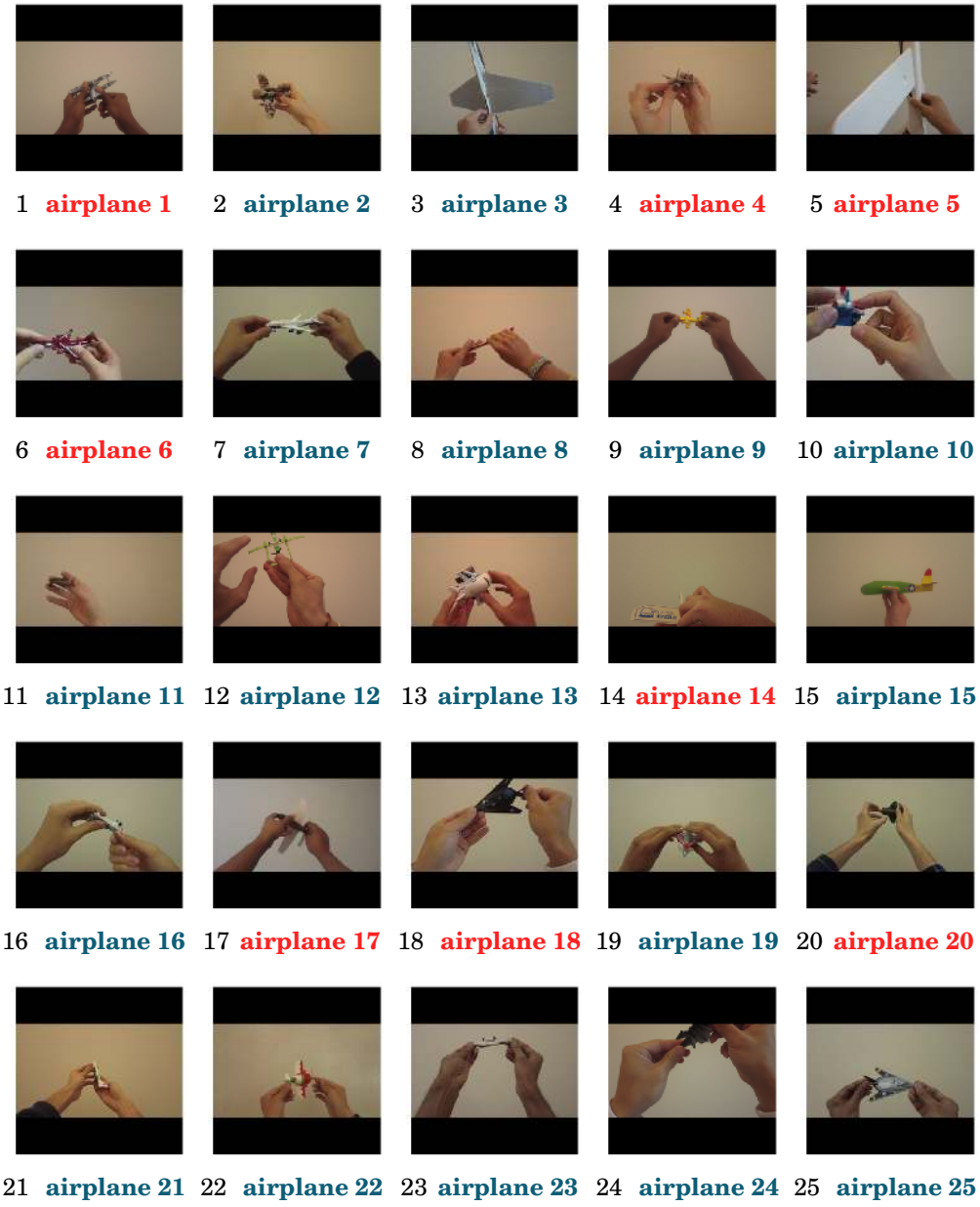


Figure A.3: Objects in ToyBox.

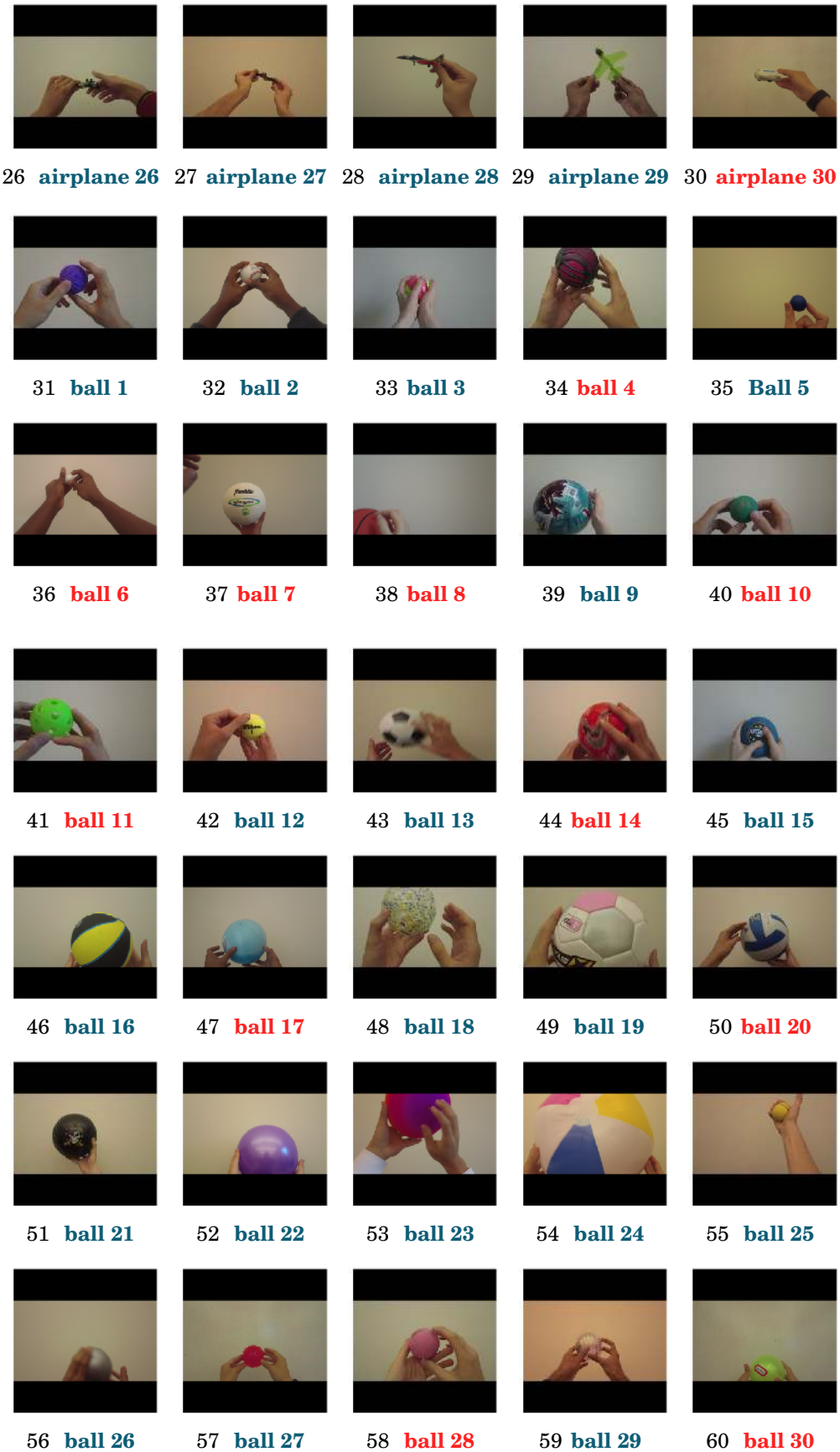


Figure A.3: Objects in ToyBox. (continuation)

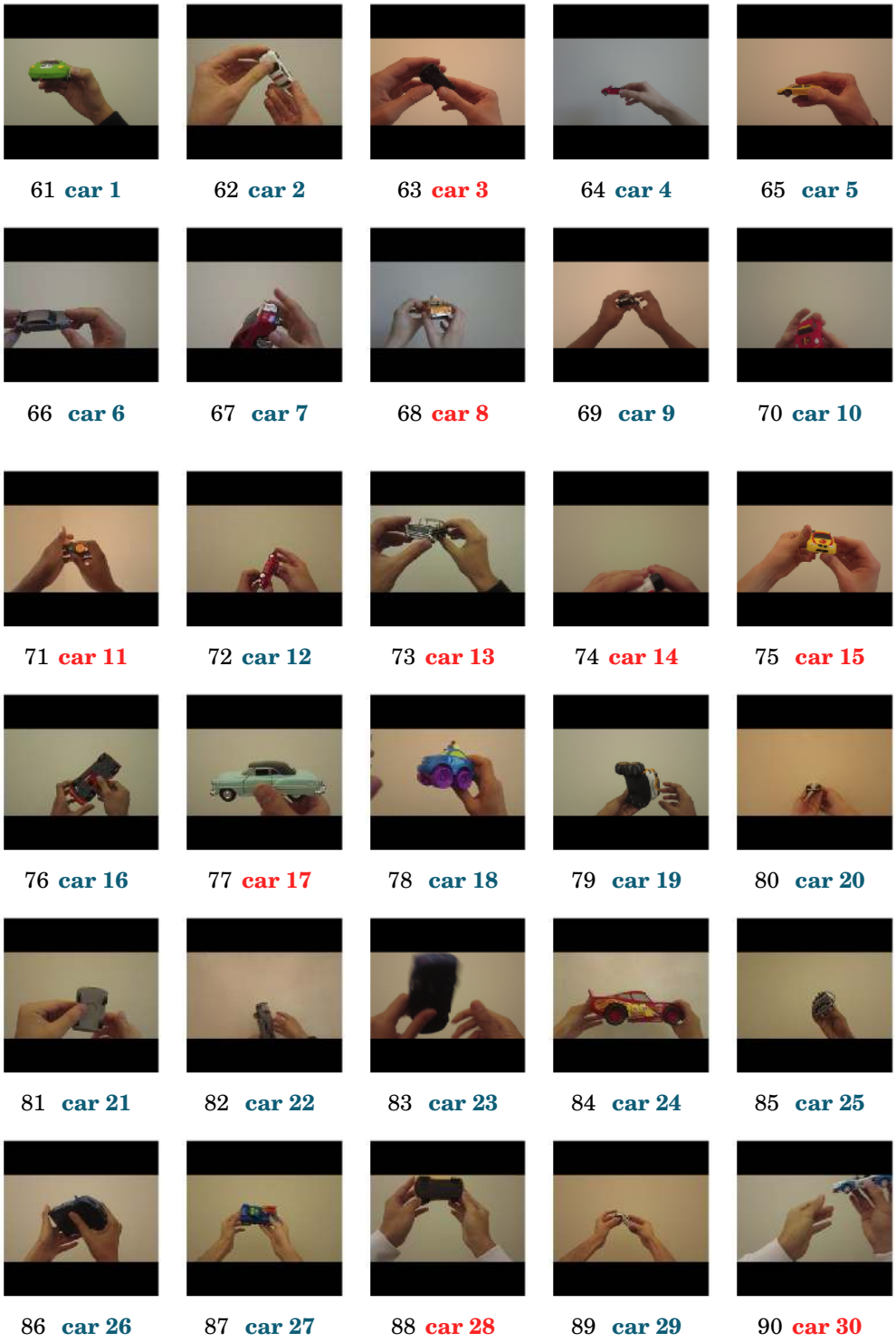


Figure A.3: Objects in ToyBox. (continuation)

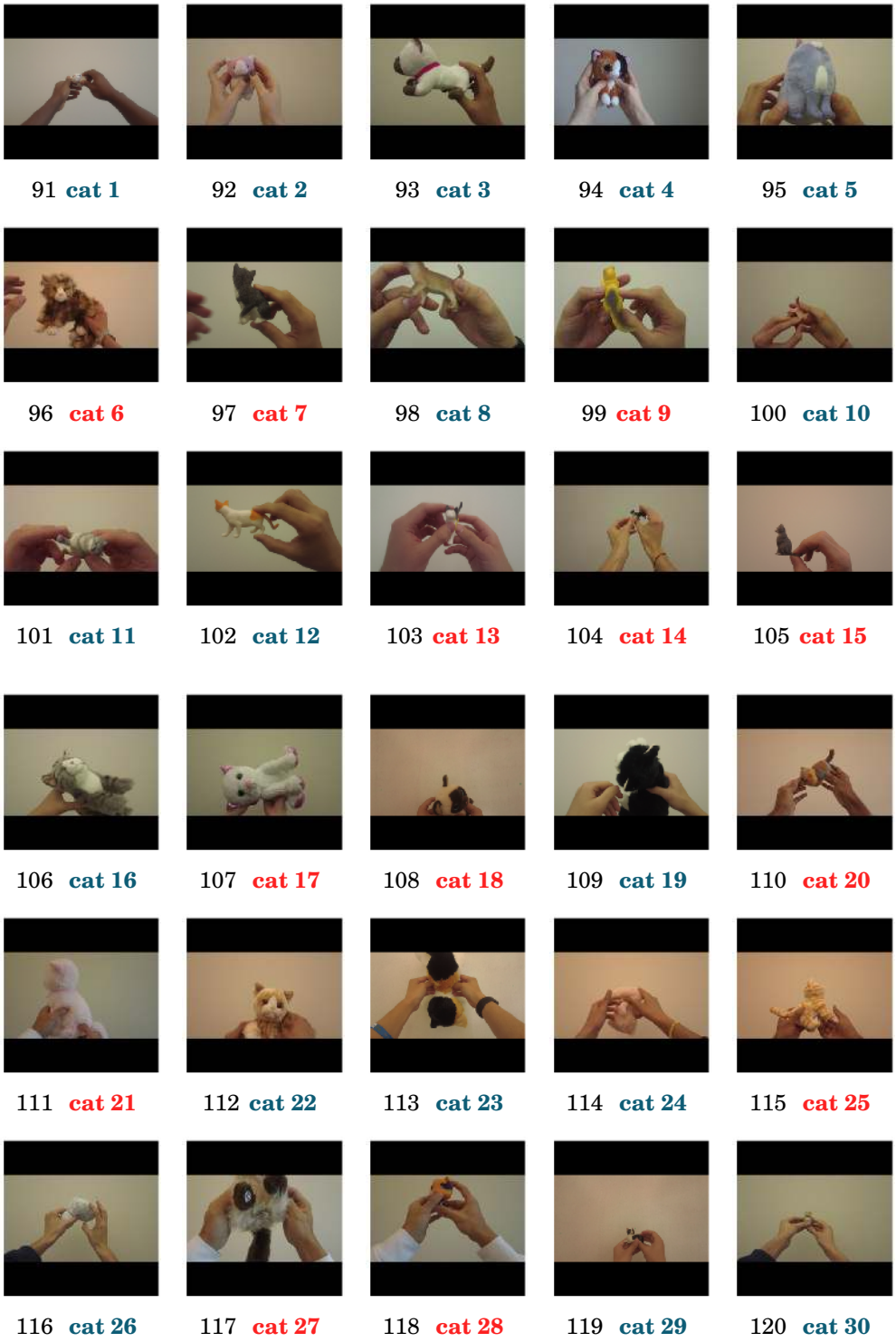


Figure A.3: Objects in ToyBox. (continuation)

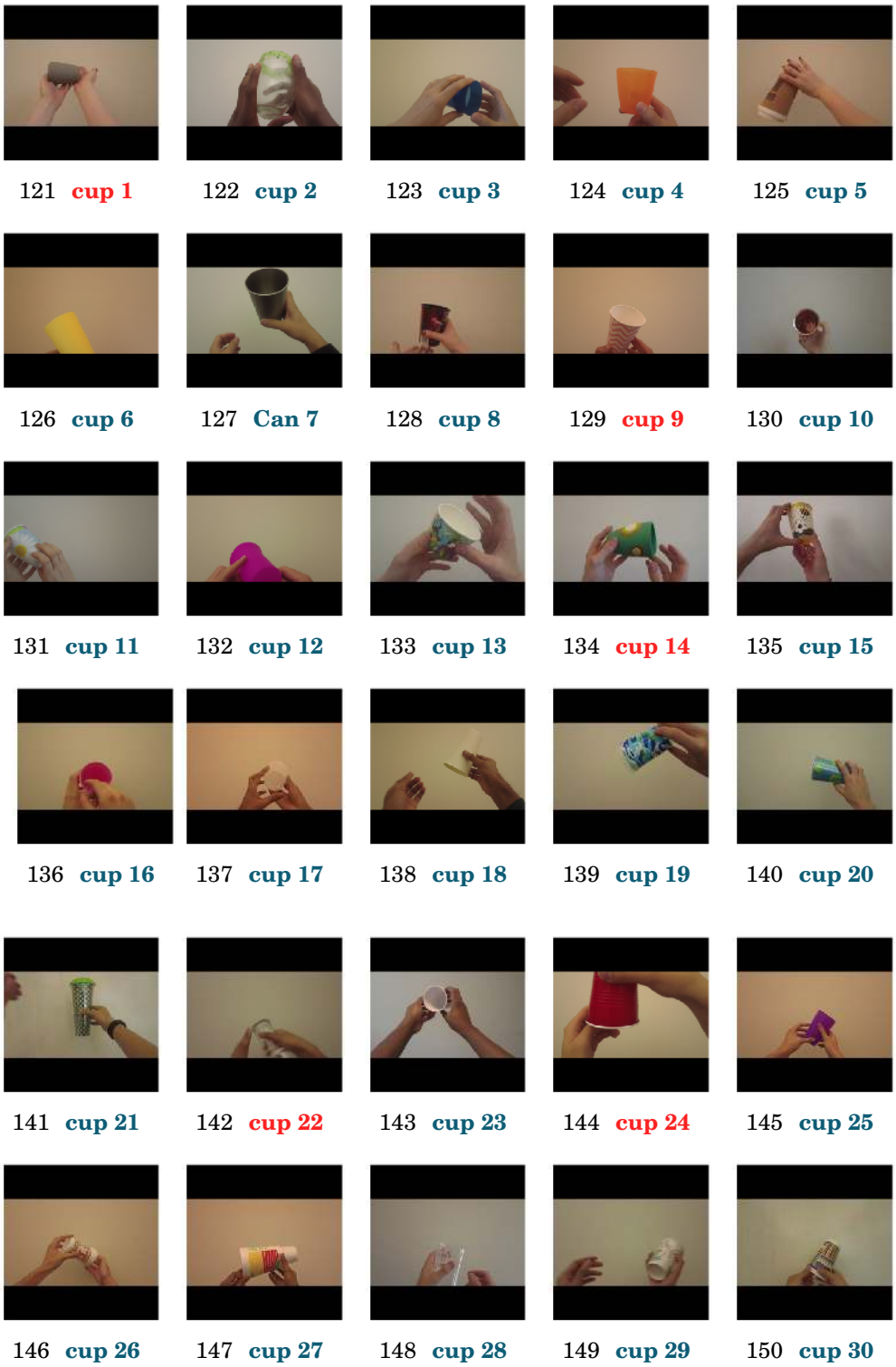


Figure A.3: Objects in ToyBox. (continuation)

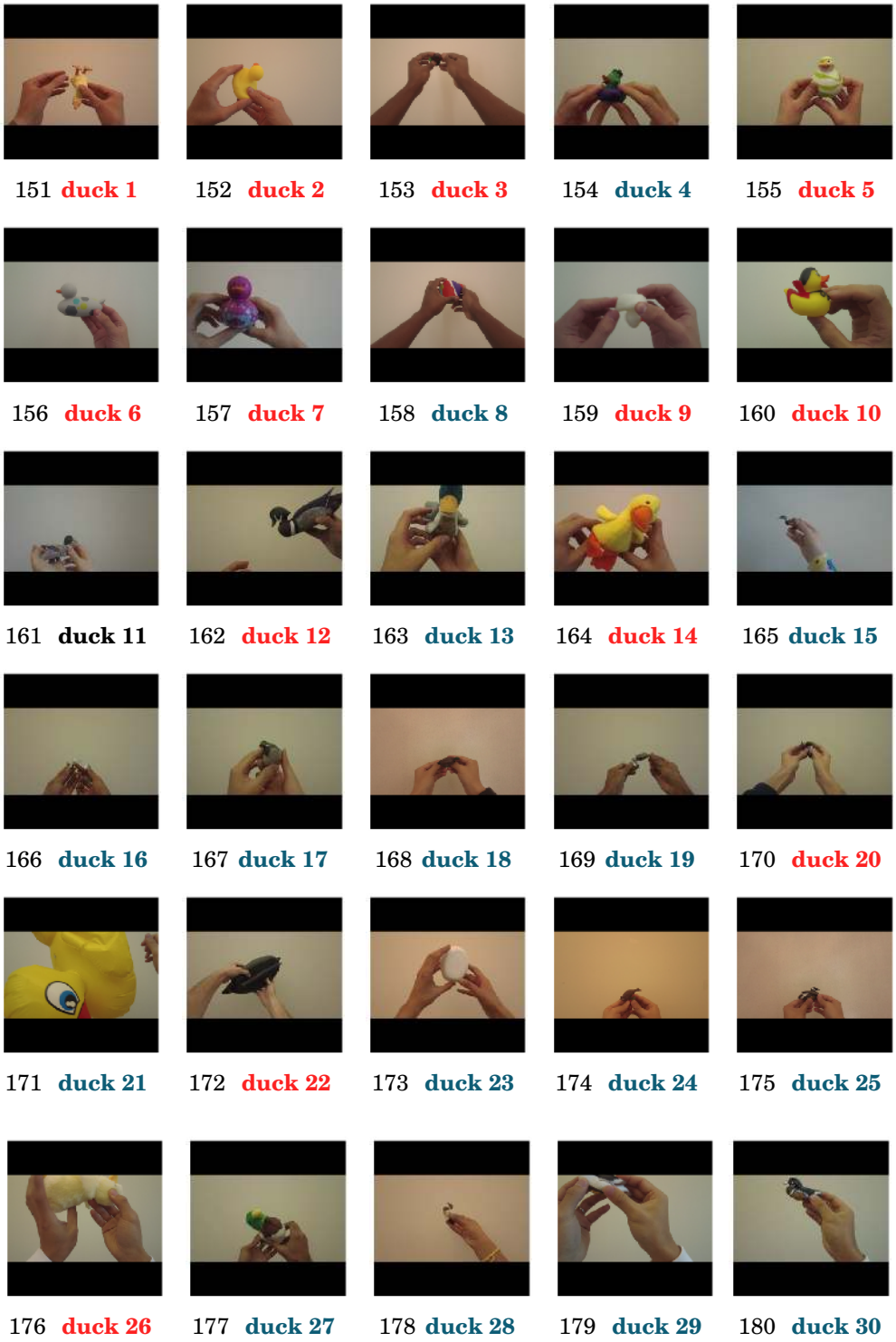


Figure A.3: Objects in ToyBox. (continuation)

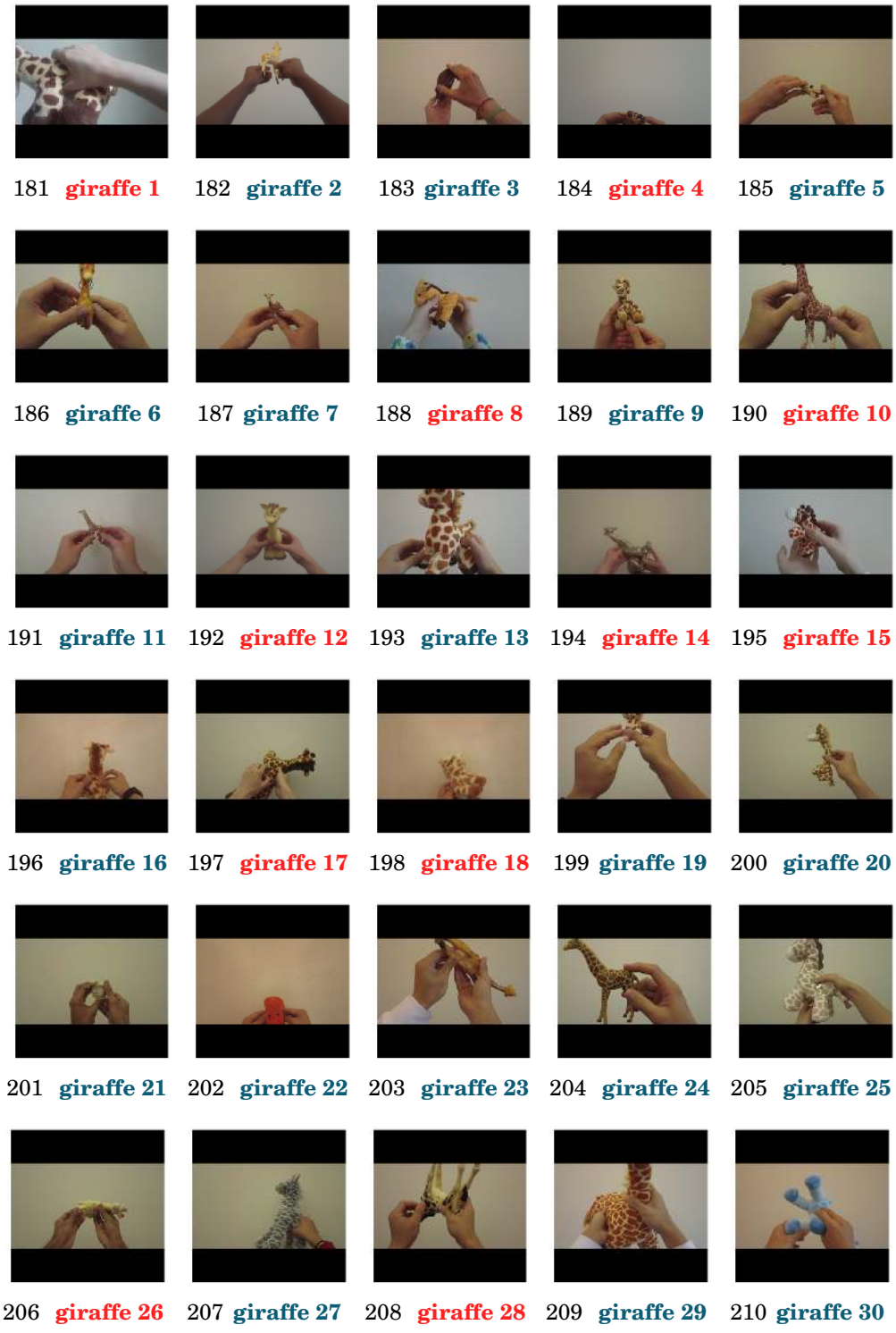


Figure A.3: Objects in ToyBox. (continuation)

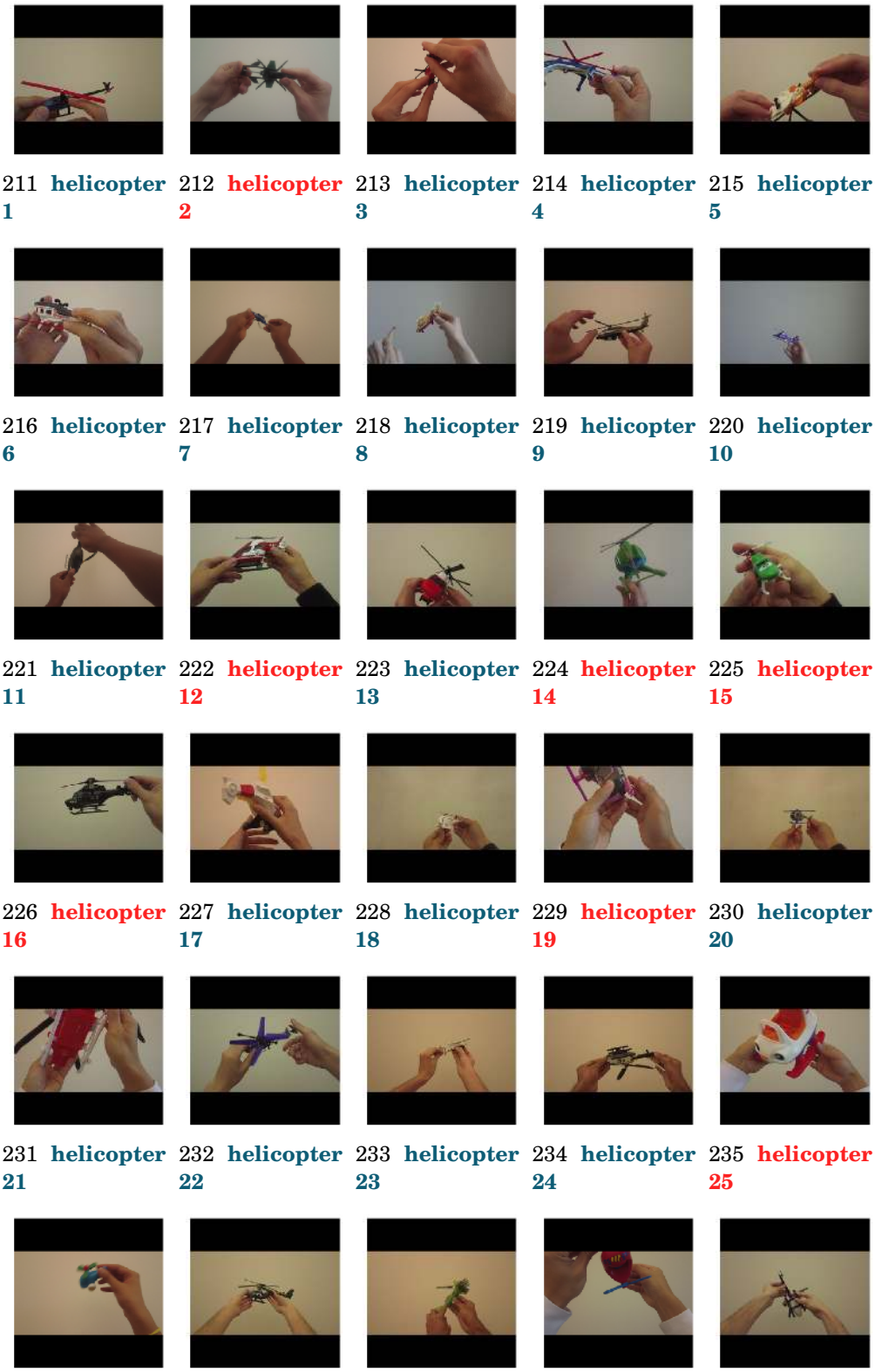


Figure A.3: Objects in ToyBox dataset. (continuation)

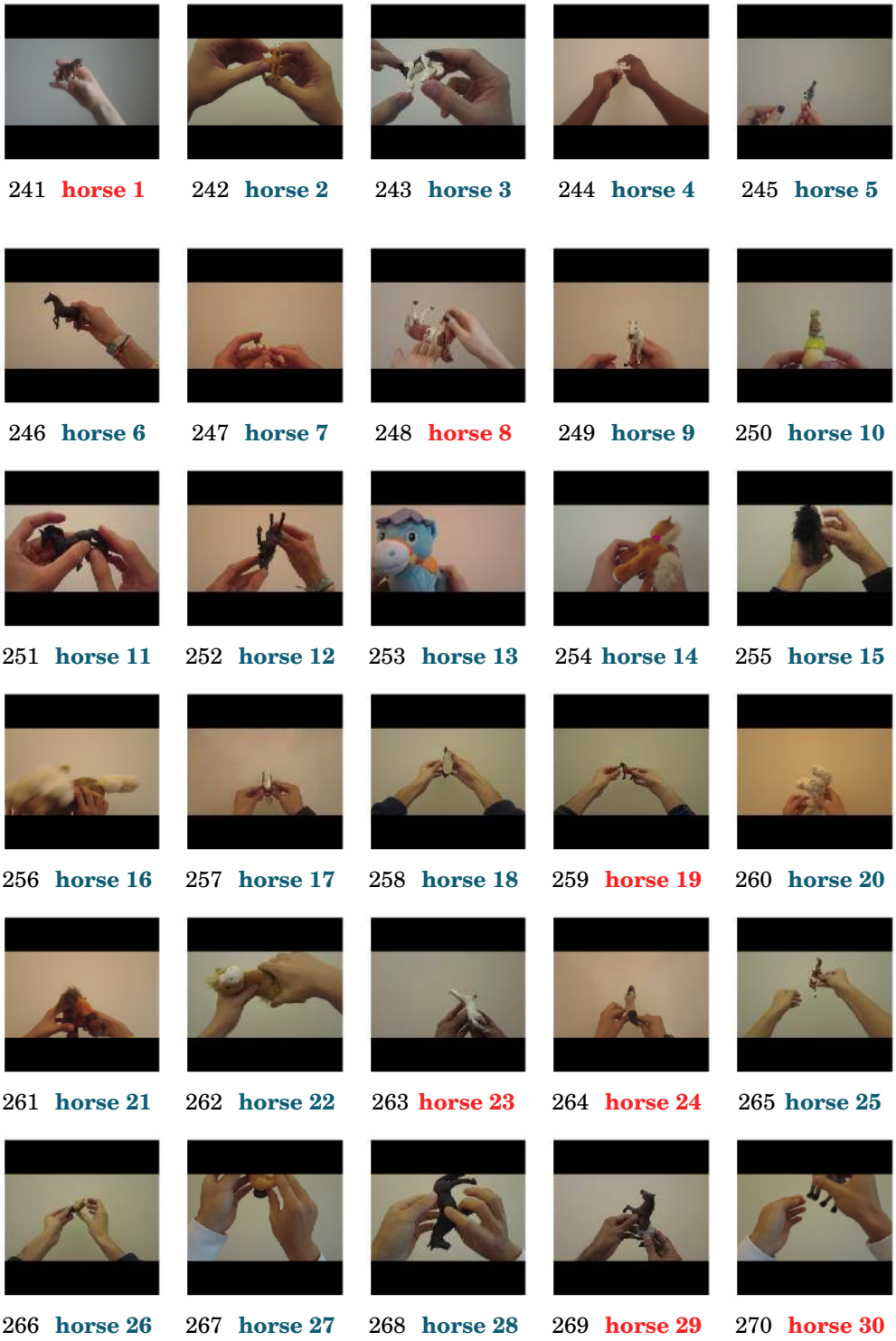


Figure A.3: Objects in ToyBox. (continuation)

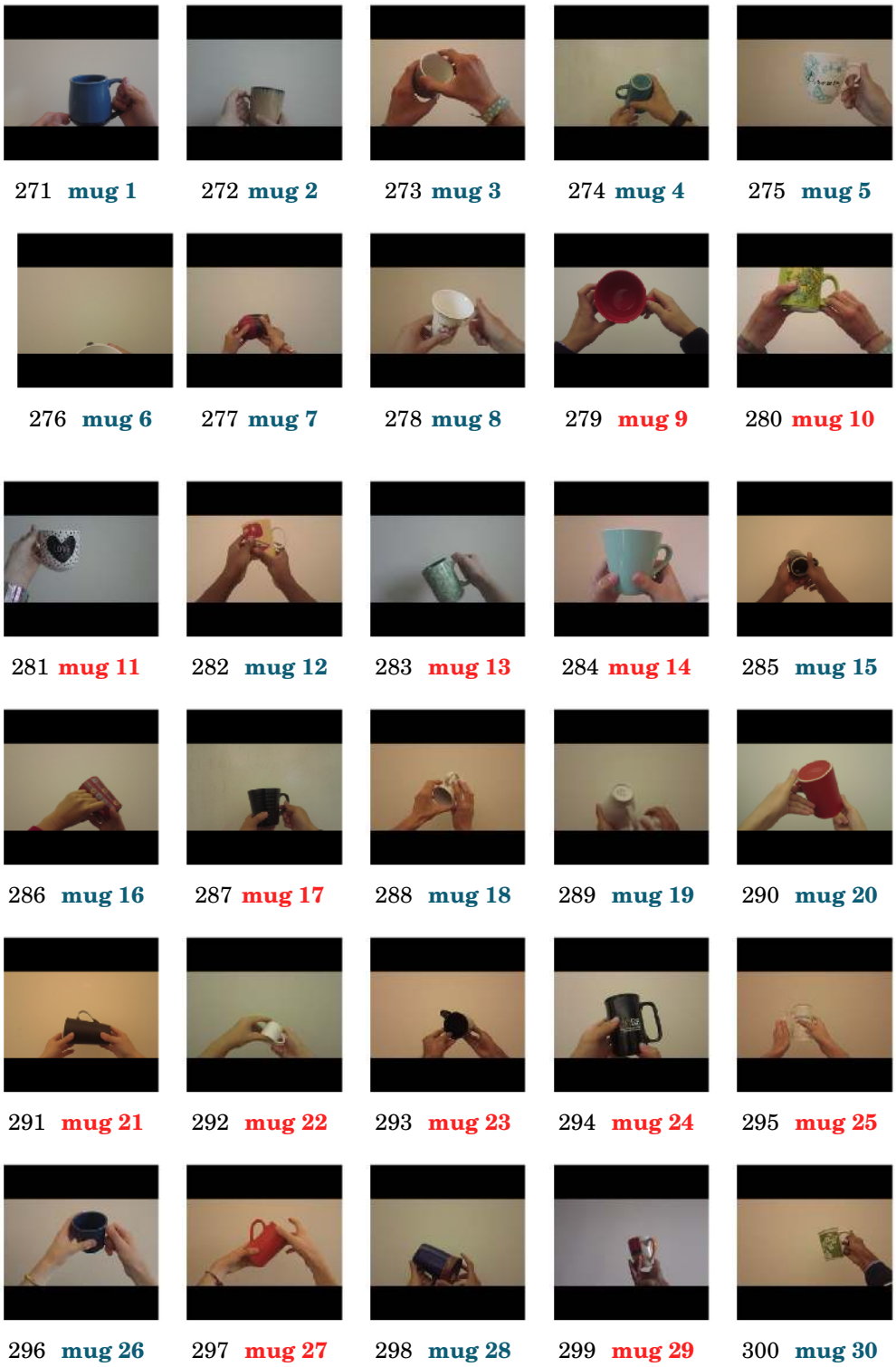


Figure A.3: Objects in ToyBox. (continuation)

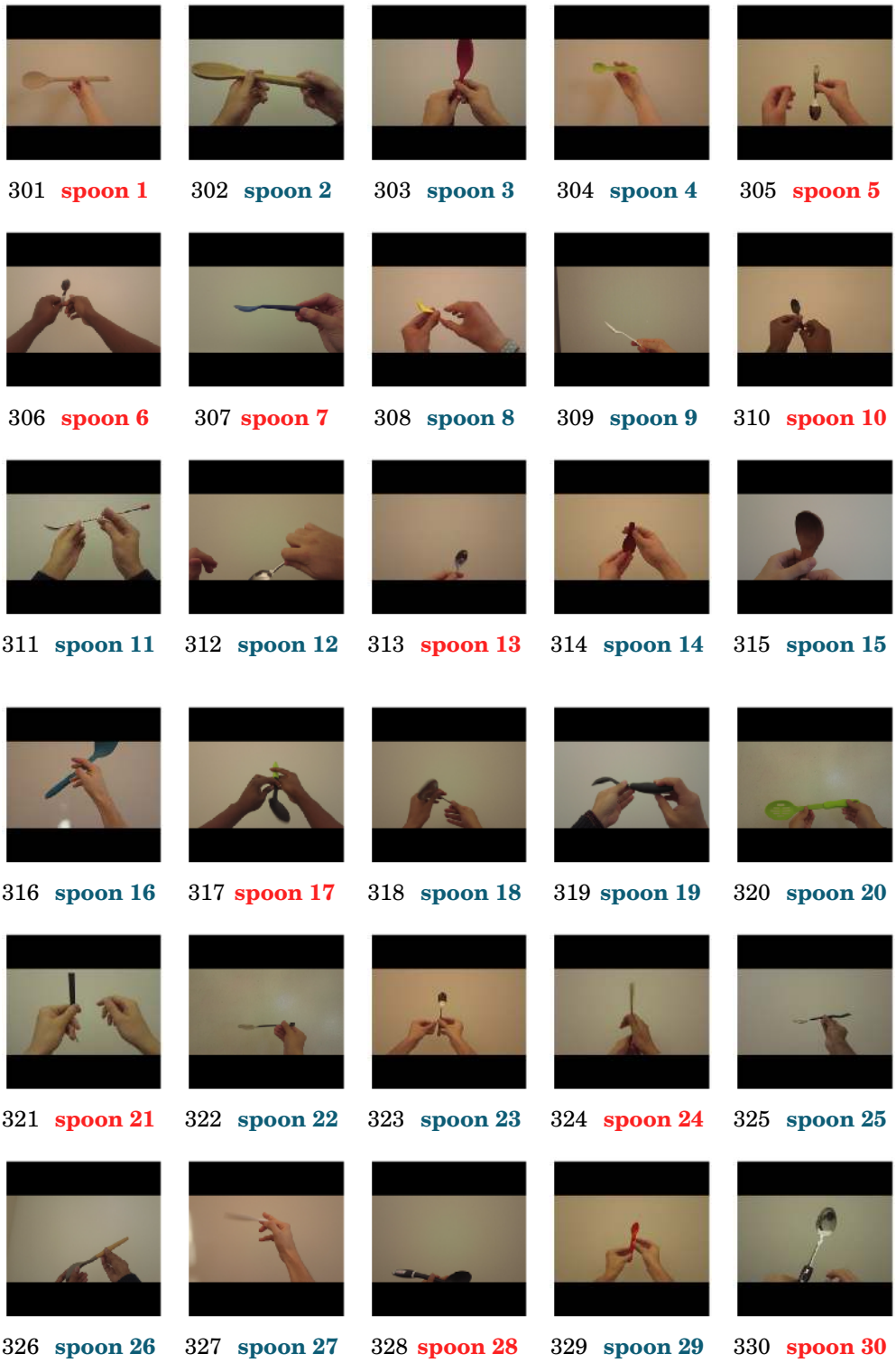


Figure A.3: Objects in ToyBox. (continuation)

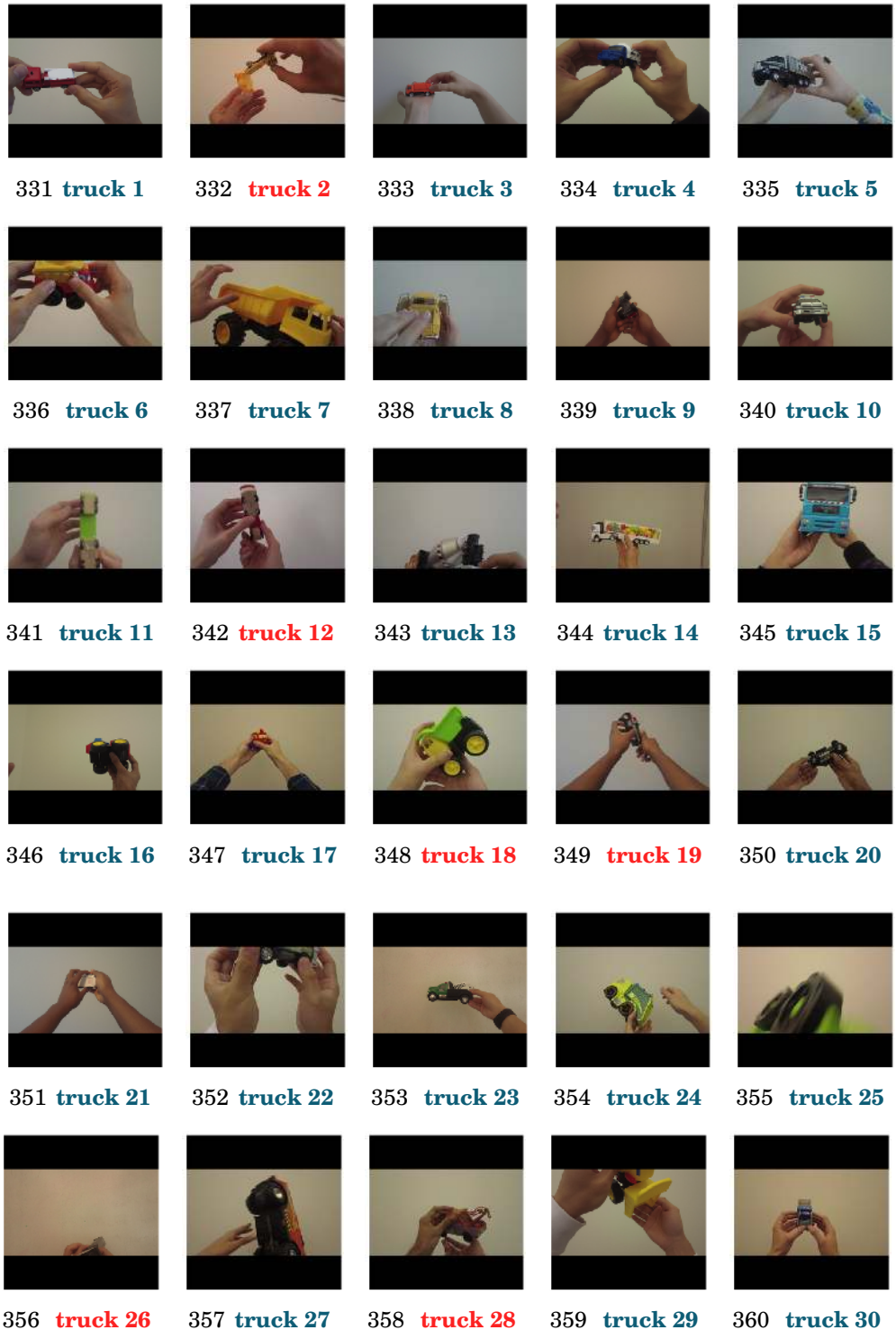


Figure A.3: Objects in ToyBox. (continuation)

A.1.4 APC-Rutgers

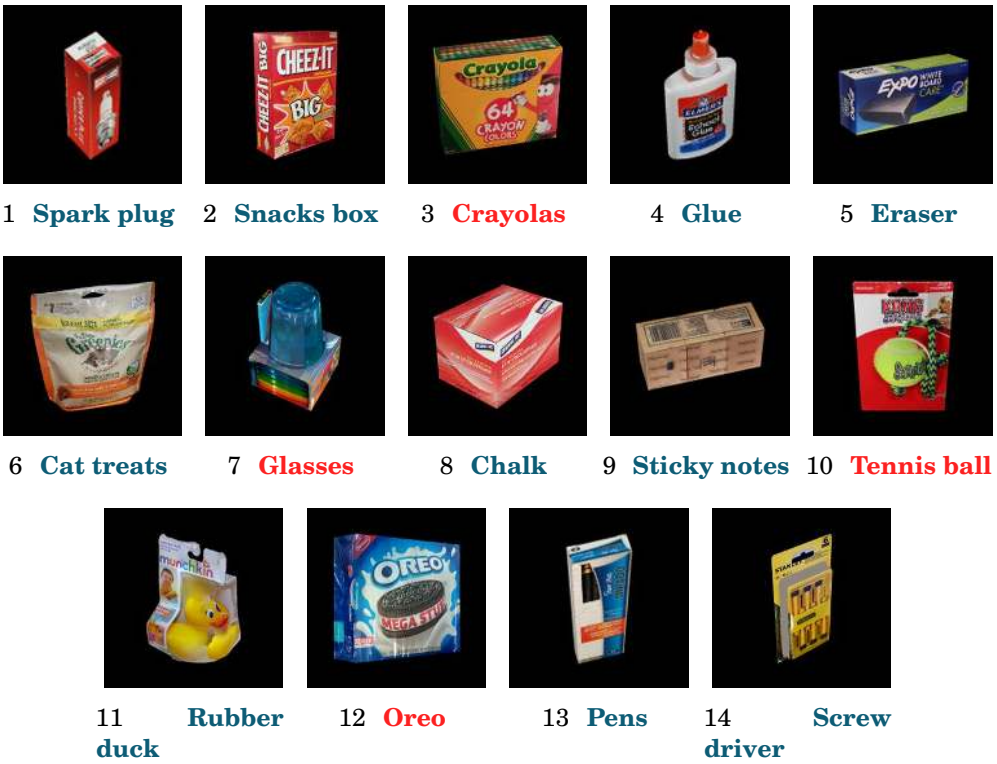


Figure A.4: Rutgers-APC Objects

A.1.5 ARC-MIT/Princeton

1 Avery Binder 2 Balloons Bottom 3 Band Aid Tape 4 Bath Sponge 5 Black Fashion Gloves

6 Burts Bees Baby Wipes 7 Cherokee Easy Tee Shirt 8 Cloud B Plush Bear 9 Colgate Toothbrush 4PK 10 Composition Book

11 Cool Shot Glue Sticks 12 Crayons 13 Creativity Chenille Stems 14 Dove Beauty Bar 15 Bottle Brush

16 Duct Tape 17 Turtle Cup 18 Elmers Glue 19 Epsom Salts 20 Expo Eraser

Figure A.5: APC-MIT/Princeton


1 **Fiskars Scissors**


2 **Flashlight**


3 **Folgers Coffee**


4 **Glue Sticks**


5 **Hand Weight**


6 **Hanes Socks**


7 **Index Cards**


8 **I Am A Bunny Book**


9 **Ice Cube Tray**


10 **Irish Spring Soap**


11 **Jane Eyre Dvd**


12 **Squeakin Eggs**


13 **Laugh Out Loud Jokes**


14 **Marbles**


15 **Measuring Spoons**


51 **Mesh Cup**


52 **Mouse Traps**


53 **Toothbrush Red**


54 **Shower Curtain**


55 **Pie Plates**


56 **Plastic Wine Glass**


57 **Platinum Dog Bowl**


58 **Spring Water**


59 **Rawlings Baseball**


60 **Reynolds Wrap**

Figure A.5: ARC (continuation)

1 Robots DVD 2 Robots Everywhere 3 Scotch Bubble Mailer 4 Scotch Sponges 5 Speed Stick

6 Staples Cards 7 Table Cloth 8 Tennis Ball Container 9 Ticonderoga Pencils 10 Tissue Box

11 Toilet Brush 12 Up Glucose Bottle 13 White Facecloth 14 Windex 15 Woods Extension Cord

Figure A.5: ARC (continuation)

A.1.6 Linemode

Figure A.6: Objects in Linemode, enumeration is consistent with legend in Figure 4.6

A.1.7 TU-Dresden Light

Figure A.7: TU-Dresden

A.1.8 Toyota-Light

Figure A.8: Objects in Toyota-Light, enumeration is consistent with legend in Figure 4.6

A.1.9 Tejani

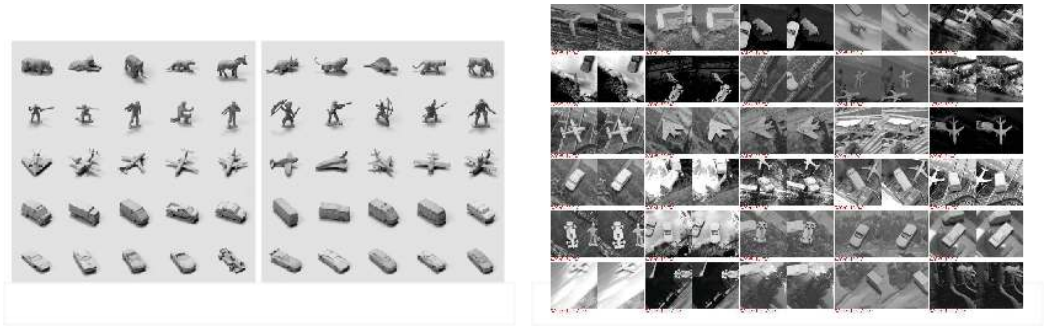
Figure A.9: Objects in Tejani, enumeration is consistent with legend in Figure 4.6

A.1.10 In-Situ Household

Figure A.10: In-situ Dataset

A.2 Training and Testing Conditions in Common Object Recognition Datasets

(a) Coil-100 (b) Big Bird



(c) Norb train (d) Norb Test

(e) ALOI Train

Figure A.11: Datasets

(k) iCub Train (l) iCub test

(m) CORe50 Train (n) CORe50 Test

(o) in-situ train (p) in-situ test

Figure A.11: Datasets (continuation)

(a) ALOI Test

(b) Intel-ego train

(c) Intel-ego test

Figure A.11: Datasets (continuation)

BIBLIOGRAPHY

Aljundi, R., Chakravarty, P., and Tuytelaars, T. (2017). Expert gate: Lifelong learning with a network of experts. In *IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, July 21-26*, pages 7120–7129.

Ba, J. and Caruana, R. (2014). Do deep nets really need to be deep? *Advances in Neural Information Processing Systems 27*, pages 2654–2662.

Bay, H., Ess, A., Tuytelaars, T., and Van Gool, L. (2008). Speeded-up robust features (surf). *Computer Vision and Image Understanding*, 110(3):346–359.

Bellet, A., Habrard, A., and Sebban, M. (2013). A survey on metric learning for feature vectors and structured data. *CoRR*, abs/1306.6709.

Bianco, S., Cadène, R., Celona, L., and Napoletano, P. (2018). Benchmark analysis of representative deep neural network architectures. *IEEE Access*, 6:64270–64277.

Borji, A., Izadi, S., and Itti, L. (2016). ilab-20m: A large-scale controlled object dataset to investigate deep learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV*, pages 2221–2230.

Bruna, J., Zaremba, W., Szlam, A., and LeCun, Y. (2014). Spectral networks and locally connected networks on graphs. In *2nd International Conference on Learning Representations, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*.

Bucher, M., Herbin, S., and Jurie, F. (2017). Generating visual representations for zero-shot classification. In *International Conference on Computer Vision Workshops*. IEEE.

Canziani, A. (2016). Cortexnet. `https://github.com/e-lab/pytorch-CortexNet`.

Canziani, A. and Culurciello, E. (2017). Cortexnet: a generic network family for robust visual temporal representations. *CoRR*, abs/1706.02735.

Castro, F. M., Marín-Jiménez, M. J., Guil, N., Schmid, C., and Alahari, K. (2018). End-to-End Incremental Learning. In Ferrari, V., Hebert, M., Sminchisescu, C., and Weiss, Y., editors, *European Conference on Computer Vision*, volume 11216 of *Lecture Notes in Computer Science*, pages 241–257, Munich, Germany. Springer.

Chatfield, K., Simonyan, K., Vedaldi, A., and Zisserman, A. (2014). Return of the devil in the details: Delving deep into convolutional nets. In *Proceedings of the British Machine Vision Conference*. BMVA Press.

Chen, W., Chen, X., Zhang, J., and Huang, K. (2017). Beyond triplet loss: A deep quadruplet network for person re-identification. *The IEEE Conference on Computer Vision and Pattern Recognition*, pages 403–412.

Cheng, G., Zhou, P., and Han, J. (2016). Learning rotation-invariant convolutional neural networks for object detection in vhr optical remote sensing images. *IEEE Transactions on Geoscience and Remote Sensing*, 54(12):7405–7415.

Chopra, S., Balakrishnan, S., and Gopalan, R. (2013). Dlid: Deep learning for domain adaptation by interpolating between domains. In *in International Conference on Machine Learning. Workshop on Challenges in Representation Learning*.

Chung, J., Gulcehre, C., Cho, K., and Bengio, Y. (2014). Empirical evaluation of gated recurrent neural networks on sequence modeling. *Advances in Neural Information Processing Systems. Workshop on Deep Learning, December 2014*.

Clark, R., Wang, S., Markham, A., Trigoni, N., and Wen, H. (2017). Vidloc: A deep spatio-temporal model for 6-dof video-clip relocalization. *The IEEE Conference on Computer Vision and Pattern Recognition*, pages 6856–6864.

Damen, D., Bunnun, P., Calway, A., and Mayol-Cuevas, W. (2012). Real-time learning and detection of 3d texture-less objects: A scalable approach. In *British Machine Vision Conference*. BMVA.

Deng, J., Guo, J., Xue, N., and Zafeiriou, S. (2019). Arcface: Additive angular margin loss for deep face recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*.

Donahue, J., Hendricks, L. A., Rohrbach, M., Venugopalan, S., Guadarrama, S., Saenko, K., and Darrell, T. (2017). Long-term recurrent convolutional networks for visual

recognition and description. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(4):677–691.

Donahue, J., Jia, Y., Vinyals, O., Hoffman, J., Zhang, N., Tzeng, E., and Darrell, T. (2014). Decaf: A deep convolutional activation feature for generic visual recognition. *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*, pages I–647–I–655.

Du, B., Xiong, W., Wu, J., Zhang, L., Zhang, L., and Tao, D. (2017). Stacked convolutional denoising auto-encoders for feature representation. *IEEE Transactions on Cybernetics*, 47(4):1017–1027.

Duan, L., Tsang, I. W., Xu, D., and Chua, T.-S. (2009). Domain adaptation from multiple sources via auxiliary classifiers. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 289–296, New York, NY, USA. ACM.

Eilertsen, G., Mantiuk, R. K., and Unger, J. (2019). Single-frame regularization for temporally stable cnns. *The IEEE Conference on Computer Vision and Pattern Recognition*.

Erhan, D., Szegedy, C., Toshev, A., and Anguelov, D. (2014). Scalable object detection using deep neural networks. *The IEEE Conference on Computer Vision and Pattern Recognition*.

Everingham, M., Van Gool, L., Williams, C. K. I., Winn, J., and Zisserman, A. (2010). The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, 88(2):303–338.

Farquhar, S. and Gal, Y. (2018). Differentially private continual learning. In *Privacy in Machine Learning and Artificial Intelligence workshop*.

Fawzi, A., Moosavi-Dezfooli, S.-M., and Frossard, P. (2016). Robustness of classifiers: from adversarial to random noise. *Advances in Neural Information Processing Systems 29*, pages 1632–1640.

Feichtenhofer, C., Pinz, A., and Wildes, R. P. (2017a). Spatiotemporal multiplier networks for video action recognition. *IEEE Conference on Computer Vision and Pattern Recognition*, pages 7445–7454.

Feichtenhofer, C., Pinz, A., and Wildes, R. P. (2017b). Temporal residual networks for dynamic scene recognition. *IEEE Conference on Computer Vision and Pattern Recognition*, pages 7435–7444.

French, R. M. (1999). Catastrophic forgetting in connectionist networks. *Trends in Cognitive Sciences*, 3:128–135.

Furnari, A., Farinella, G. M., and Battiato, S. (2017). Recognizing personal locations from egocentric videos. *IEEE Transactions on Human-Machine Systems*, 47(1):6–18.

Ganin, Y., Ustinova, E., Ajakan, H., Germain, P., Larochelle, H., Laviolette, F., Marchand, M., and Lempitsky, V. (2016). Domain-adversarial training of neural networks. *Journal of Machine Learning Research*, 17(1):2096–2030.

Gecer, B., Balntas, V., and Kim, T. (2017). Learning deep convolutional embeddings for face representation using joint sample- and set-based supervision. pages 1665–1672.

Geusebroek, J. M., Burghouts, G. J., and Smeulders, A. W. M. (2005). The amsterdam library of object images. *International Journal of Computer Vision*, 61(1):103–112.

Ghifary, M. (2016). *"Domain Adaptation and Domain Generalization with Representation Learning"*. PhD thesis, Victoria University of Wellington.

Gidaris, S. and Komodakis, N. (2018). Dynamic few-shot visual learning without forgetting. In *The IEEE Conference on Computer Vision and Pattern Recognition*.

Girshick, R. (2015). Fast r-cnn. In *Proceedings of the 2015 IEEE International Conference on Computer Vision*, pages 1440–1448, Washington, DC, USA. IEEE Computer Society.

Girshick, R., Donahue, J., Darrell, T., and Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 580–587, Washington, DC, USA. IEEE Computer Society.

Göhring, D., Judy, H., Erik, R., Kate, S., and Trevor, D. (2014). Interactive adaptation of real-time object detectors. *2014 IEEE International Conference on Robotics and Automation*, pages 1282–1289.

Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press. `http://www.deeplearningbook.org`.

Graves, A., Jaitly, N., and Mohamed, A. (2013). Hybrid speech recognition with deep bidirectional LSTM. In *2013 IEEE Workshop on Automatic Speech Recognition and Understanding, Olomouc, Czech Republic, December 8-12, 2013*, pages 273–278.

Hadsell, R., Chopra, S., and LeCun, Y. (2006). Dimensionality reduction by learning an invariant mapping. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2, pages 1735–1742.

Hall, D., Dayoub, F., Skinner, J., Corke, P., Carneiro, G., and Sünderhauf, N. (2018). Probability-based detection quality (PDQ): A probabilistic approach to detection evaluation. *CoRR*, abs/1811.10800.

Hara, K., Kataoka, H., and Satoh, Y. (2017). Learning spatio-temporal features with 3d residual networks for action recognition. *2017 IEEE International Conference on Computer Vision Workshops*, pages 3154–3160.

He, K., Gkioxari, G., Dollár, P., and Girshick, R. (2017). Mask R-CNN. *IEEE International Conference on Computer Vision*, pages 2980–2988.

He, K., Zhang, X., Ren, S., and Sun, J. (2015). Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1026–1034, Washington, DC, USA. IEEE Computer Society.

He, K., Zhang, X., Ren, S., and Sun, J. (2015). Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(9):1904–1916.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition Las Vegas, NV, USA, June 27-30*, pages 770–778.

He, X., Zhou, Y., Zhou, Z., Bai, S., and Bai, X. (2018). Triplet-center loss for multi-view 3d object retrieval. *The IEEE Conference on Computer Vision and Pattern Recognition*.

Hermans, A., Beyer, L., and Leibe, B. (2017). In defense of the triplet loss for person re-identification. *CoRR*, abs/1703.07737.

Hinterstoisser, S., Lepetit, V., Ilic, S., Holzer, S., Bradski, G., Konolige, K., and Navab, N. (2013). Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes. In Lee, K. M., Matsushita, Y., Rehg, J. M., and Hu, Z., editors, *Asian Conference on Computer Vision*, pages 548–562, Berlin, Heidelberg. Springer Berlin Heidelberg.

Hinton, G., Vinyals, O., and Dean, J. (2015). Distilling the knowledge in a neural network. *Conference on Neural Information Processing Systems 28. Deep Learning and Representation Learning Workshop*.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Hodaň, T., Damen, D., Mayol-Cuevas, W., and Matas, J. (2015). Efficient texture-less object detection for augmented reality guidance. *IEEE International Symposium on Mixed and Augmented Reality Workshops*.

Hodaň, T., Haluza, P., Obdržálek, Š., Matas, J., Lourakis, M., and Zabulis, X. (2017). T-LESS: An RGB-D dataset for 6D pose estimation of texture-less objects. *IEEE Winter Conference on Applications of Computer Vision*.

Hodan, T., Michel, F., Brachmann, E., Kehl, W., Buch, A. G., Kraft, D., Drost, B., Vidal, J., Ihrke, S., Zabulis, X., Sahin, C., Manhardt, F., Tombari, F., Kim, T., Matas, J., and Rother, C. (2018). BOP: benchmark for 6d object pose estimation. In *European Conference on Computer Vision*, volume 11214 of *Lecture Notes in Computer Science*, pages 19–35. Springer.

Hosseini, H. and Poovendran, R. (2017). Deep neural networks do not recognize negative images. *CoRR*, abs/1703.06857.

Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., and Adam, H. (2017). Mobilenets: Efficient convolutional neural networks for mobile vision applications. *CoRR*, abs/1704.04861.

Huang, C., Li, Y., Loy, C. C., and Tang, X. (2016a). Learning deep representation for imbalanced classification. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 5375–5384.

Huang, C., Loy, C. C., and Tang, X. (2016b). Local similarity-aware deep feature embedding. *Advances in Neural Information Processing Systems 29*, pages 1262–1270.

Huang, G., Liu, Z., van der Maaten, L., and Weinberger, K. Q. (2017a). Densely connected convolutional networks. *The IEEE Conference on Computer Vision and Pattern Recognition*.

Huang, J., Rathod, V., Sun, C., Zhu, M., Korattikara, A., Fathi, A., Fischer, I., Wojna, Z., Song, Y., Guadarrama, S., and Murphy, K. (2017b). Speed/accuracy trade-offs for modern convolutional object detectors. In *The IEEE Conference on Computer Vision and Pattern Recognition*.

Ioffe, S. and Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. *Proceedings of the 32Nd International Conference on International Conference on Machine Learning - Volume 37*, pages 448–456.

Jain, L. C., Seera, M., Lim, C. P., and Balasubramaniam, P. (2014). A review of online learning in supervised neural networks. *Neural Computing and Applications*, 25(3):491–509.

Jarrett, K., Kavukcuoglu, K., Ranzato, M., and LeCun, Y. (2009). What is the best multi-stage architecture for object recognition? In *International Conference on Computer Vision*, pages 2146–2153. IEEE.

Ji, S., Xu, W., Yang, M., and Yu, K. (2013). 3D Convolutional Neural Networks for Human Action Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(1):221–231.

Junh, Z. (2019). Disentangled representation learning gan for pose-invariant face recognition. `https://github.com/zhangjunh/DR-GAN-by-pytorch`.

Karpathy, A., Toderici, G., Shetty, S., Leung, T., Sukthankar, R., and Fei-Fei, L. (2014). Large-scale video classification with convolutional neural networks. In *Conference on Computer Vision and Pattern Recognition*.

Kashif, M. N., Raza, S. E. A., Sirinukunwattana, K., Arif, M., and Rajpoot, N. (2016). Handcrafted features with convolutional neural networks for detection of tumor cells in histology images. In *2016 IEEE 13th International Symposium on Biomedical Imaging*, pages 1029–1032.

Kehl, W., Manhardt, F., Tombari, F., Ilic, S., and Navab, N. (2017). SSD-6D: Making rgb-based 3d detection and 6d pose estimation great again. In *The IEEE International Conference on Computer Vision*.

Kemaev, I., Polykovskiy, D., and Vetrov, D. (2018). Reset: Learning recurrent dynamic routing in resnet-like neural networks. In Zhu, J. and Takeuchi, I., editors, *Proceedings*

of The 10th Asian Conference on Machine Learning, volume 95 of *Proceedings of Machine Learning Research*, pages 422–437. PMLR.

Koch, G., Zemel, R., and Salakhutdinov, R. (2015). Siamese neural networks for one-shot image recognition. In *International Conference on Machine Learning. Deep Learning Workshop*.

Koubaroulis, D., Matas, J., and Kittler, J. (2002). Evaluating colour-based object recognition algorithms using the soil-47 database. In *in Asian Conference on Computer Vision*, pages 840–845.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In Pereira, F., Burges, C. J. C., Bottou, L., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc.

Kumar, B. G. V., Carneiro, G., and Reid, I. D. (2015). Learning local image descriptors with deep siamese and triplet convolutional networks by minimising global loss functions. *The IEEE Conference on Computer Vision and Pattern Recognition*, abs/1512.09272.

Kumar, B. G. V., Harwood, B., Carneiro, G., Reid, I. D., and Drummond, T. (2017). Smart mining for deep metric learning. *International Conference on Computer Vision*, abs/1704.01285.

Käding, C., Rodner, E., Freytag, A., and Denzler, J. (2016). Fine-tuning deep neural networks in continuous learning scenarios. In *Asian Conference on Computer Vision. Workshop on Interpretation and Visualization of Deep Neural Nets*.

Lagunes-Fortiz, M. (2019a). Pytorch implementation of center loss. `https://github.com/MikeLagunes/ste-network`.

Lagunes-Fortiz, M. (2019b). Pytorch implementation of centroids triplet loss. `https://github.com/MikeLagunes?tab=repositories`.

Lagunes-Fortiz, M., Damen, D., and Mayol, W. (2018). Learning discriminative embeddings for object reconition on-the-fly. In *IEEE International Conference on Robotics and Automation*.

Lai, K., Bo, L., Ren, X., and Fox, D. (2011). A large-scale hierarchical multi-view rgb-d object dataset. *IEEE International Conference on Robotics and Automation*, pages 1817–1824.

Lake, B. M., Ullman, T. D., Tenenbaum, J. B., and Gershman, S. J. (2017). Building machines that learn and think like people. *Behavioral and Brain Sciences*, 40:e253.

Lan, Z., Yu, S., Yao, D., Lin, M., Raj, B., and Hauptmann, A. (2016). The best of bothworlds: Combining data-independent and data-driven approaches for action recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 1196–1205.

Le, L., Patterson, A., and White, M. (2018). Supervised autoencoders: Improving generalization performance with unsupervised regularizers. In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R., editors, *Advances in Neural Information Processing Systems 31*, pages 107–117. Curran Associates, Inc.

Le, T. and Duan, Y. (2018). Pointgrid: A deep network for 3d shape understanding. In *The IEEE Conference on Computer Vision and Pattern Recognition*.

LeCun, Y., Huang, F. J., and Bottou, L. (2004). Learning methods for generic object recognition with invariance to pose and lighting. In *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 97–104, Washington, DC, USA. IEEE Computer Society.

Li, H., Jialin Pan, S., Wang, S., and Kot, A. C. (2018a). Domain generalization with adversarial feature learning. In *The IEEE Conference on Computer Vision and Pattern Recognition*.

Li, Y., Zhang, J., Zhang, J., and Huang, K. (2018b). Discriminative learning of latent features for zero-shot recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition*.

Li, Z., Gavves, E., Jain, M., and Snoek, C. G. M. (2016). Videolstm convolves, attends and flows for action recognition. *In The IEEE Conference on Computer Vision and Pattern Recognition*, abs/1607.01794.

Li, Z. and Hoiem, D. (2016). Learning without forgetting. *In European Conference on Computer Vision*, abs/1606.09282.

Lin, M., Chen, Q., and Yan, S. (2013). Network in network. *CoRR*, abs/1312.4400.

Lin, T., Goyal, P., Girshick, R. B., He, K., and Dollár, P. (2018). Focal loss for dense object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, abs/1708.02002.

Liu, B., Wang, X., Dixit, M., Kwitt, R., and Vasconcelos, N. (2018a). Feature space transfer for data augmentation. In *The IEEE Conference on Computer Vision and Pattern Recognition*.

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., and Berg, A. C. (2016a). SSD: Single shot multibox detector. *14th European Conference on Computer Vision, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I*, pages 21–37.

Liu, W., Wen, Y., Yu, Z., and Yang, M. (2016b). Large-margin softmax loss for convolutional neural networks. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, pages 507–516. JMLR.org.

Liu, Y., Yeh, Y., Fu, T., Wang, S., Chiu, W., and Wang, Y. F. (2018b). Detach and adapt: Learning cross-domain disentangled deep representation. In *IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, June 18-22, 2018*, pages 8867–8876.

Liu, Z., Li, J., Shen, Z., Huang, G., Yan, S., and Zhang, C. (2017). Learning efficient convolutional networks through network slimming. In *International Conference on Computer Vision*.

Lomonaco, V. and Maltoni, D. (2017). Core50: a new dataset and benchmark for continuous object recognition. In *Proceedings of the 1st Annual Conference on Robot Learning*, volume 78 of *Proceedings of Machine Learning Research*, pages 17–26. PMLR.

Long, M., CAO, Z., Wang, J., and Jordan, M. I. (2018). Conditional adversarial domain adaptation. In *Advances in Neural Information Processing Systems 31*, pages 1647–1657. Curran Associates, Inc.

Lopez-Paz, D. and Ranzato, M. A. (2017). Gradient episodic memory for continual learning. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R., editors, *Advances in Neural Information Processing Systems 30*, pages 6467–6476. Curran Associates, Inc.

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110.

Maiettini, E., Pasquale, G., Rosasco, L., and Natale, L. (2017). Interactive data collection for deep learning object detectors on humanoid robots. *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*, pages 862–868.

Margolis, E. & Laurence, S. (2015). *The Conceptual Mind: New Directions in the Study of Concepts*. Cambridge: The MIT Press.

Merler, M., C, G., and S, B. (2007). Recognizing groceries in situ using in vitro training data. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8.

Milan, A., Pham, T., Vijay, K., Morrison, D., Tow, A. W., Liu, L., Erskine, J., Grinover, R., Gurman, A., Hunn, T., Kelly-Boxall, N., Lee, D., McTaggart, M., Rallos, G., Razjigaev, A., Rowntree, T., Shen, T., Smith, R., Wade-McCue, S., Zhuang, Z., Lehnert, C. F., Lin, G., Reid, I. D., Corke, P. I., and Leitner, J. (2017). Semantic segmentation from limited training data. *CoRR*, abs/1709.07665.

Miller, D., Dayoub, F., Milford, M., and Sünderhauf, N. (2019). Evaluating merging strategies for sampling-based uncertainty techniques in object detection. *International Conference on Robotics and Automation*, abs/1809.06006.

Mishra, A., Krishna Reddy, S., Mittal, A., and Murthy, H. A. (2018). A generative model for zero shot learning using conditional variational autoencoders. In *The IEEE Conference on Computer Vision and Pattern Recognition Workshops*.

Mousavian, A., Anguelov, D., Flynn, J., and Kosecka, J. (2017). 3d bounding box estimation using deep learning and geometry. *In IEEE Conference on Computer Vision and Pattern Recognition*, abs/1612.00496.

Nanni, L., Ghidoni, S., and Brahnam, S. (2017). Handcrafted vs non-handcrafted features for computer vision classification. *Pattern Recognition*, 71:158–172.

Nene, S. A., Nayar, S. K., and Murase, H. (1996). object image library coil-100. Technical report.

Ng, J. Y., Hausknecht, M. J., Vijayanarasimhan, S., Vinyals, O., Monga, R., and Toderici, G. (2015). Beyond short snippets: Deep networks for video classification. *Conference on Computer Vision and Pattern Recognition*, abs/1503.08909.

Nguyen, C. V., Li, Y., Bui, T. D., and Turner, R. E. (2018). Variational continual learning. In *International Conference on Learning Representations*.

Ning, G., Zhang, Z., Huang, C., He, Z., Ren, X., and Wang, H. (2017). Spatially supervised recurrent convolutional neural networks for visual object tracking. *IEEE International Symposium on Circuits and Systems*, abs/1607.05781.

Opitz, M., Waltner, G., Possegger, H., and Bischof, H. (2018). Deep metric learning with BIER: boosting independent embeddings robustly. *International Conference on Computer Vision*, abs/1801.04815:5189–5198.

Osherov, E. and Lindenbaum, M. (2017). Increasing cnn robustness to occlusions by reducing filter support. In *The IEEE International Conference on Computer Vision*.

Pahde, F., Nabi, M., Klein, T., and Jahnichen, P. (2018). Discriminative hallucination for multi-modal few-shot learning. In *2018 25th IEEE International Conference on Image Processing*, pages 156–160.

Pasquale, G., Ciliberto, C., Rosasco, L., and Natale, L. (2016a). Object identification from few examples by improving the invariance of a deep convolutional neural network. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4904–4911.

Pasquale, G., Ciliberto, C., Rosasco, L., and Natale, L. (2016b). Object identification from few examples by improving the invariance of a deep convolutional neural network. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4904–4911.

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., and Lerer, A. (2017). Automatic differentiation in pytorch. In *Conference on Neural Information Processing Systems Workshops*.

Pathak, D., Girshick, R. B., Dollár, P., Darrell, T., and Hariharan, B. (2017). Learning features by watching objects move. *In the IEEE Conference on Computer Vision and Pattern Recognition*, abs/1612.06370.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Pinheiro, P. O. (2018). Unsupervised domain adaptation with similarity learning. In *The IEEE Conference on Computer Vision and Pattern Recognition*.

Pinheiro, P. O., Collobert, R., and Dollar, P. (2015). Learning to segment object candidates. In Cortes, C., Lawrence, N. D., Lee, D. D., Sugiyama, M., and Garnett, R., editors,

Advances in Neural Information Processing Systems 28, pages 1990–1998. Curran Associates, Inc.

Poggio, T., Mhaskar, H., Rosasco, L., Miranda, B., and Liao, Q. (2017). Why and when can deep-but not shallow-networks avoid the curse of dimensionality: A review. *International Journal of Automation and Computing*, 14(5):503–519.

Poirson, P., Ammirato, P., Fu, C., Liu, W., Kosecka, J., and Berg, A. C. (2016). Fast single shot detection and pose estimation. *CoRR*, abs/1609.05590.

Prémont-Schwarz, I., Ilin, A., Hao, T., Rasmus, A., Boney, R., and Valpola, H. (2017). Recurrent ladder networks. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R., editors, *Advances in Neural Information Processing Systems 30*, pages 6009–6019. Curran Associates, Inc.

Rasmus, A., Berglund, M., Honkala, M., Valpola, H., and Raiko, T. (2015). Semi-supervised learning with ladder networks. In Cortes, C., Lawrence, N. D., Lee, D. D., Sugiyama, M., and Garnett, R., editors, *Advances in Neural Information Processing Systems 28*, pages 3546–3554. Curran Associates, Inc.

Rastegari, M., Ordonez, V., Redmon, J., and Farhadi, A. (2017). Xnor-net: Imagenet classification using binary convolutional neural networks. *European Conference on Computer Vision*, abs/1603.05279.

Rebuffi, S., Kolesnikov, A., Sperl, G., and Lampert, C. H. (2017). icarl: Incremental classifier and representation learning. In *IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, July 21-26*, pages 5533–5542.

Redmon, J., Divvala, S. K., Girshick, R. B., and Farhadi, A. (2016). You only look once: Unified, real-time object detection. *Conference on Computer Vision and Pattern Recognition*, abs/1506.02640.

Redmon, J. and Farhadi, A. (2016). YOLO9000: better, faster, stronger. *Conference on Computer Vision and Pattern Recognition*, abs/1612.08242.

Ren, M., Triantafillou, E., Ravi, S., Snell, J., Swersky, K., Tenenbaum, J. B., Larochelle, H., and Zemel, R. S. (2018). Meta-learning for semi-supervised few-shot classification. In *International Conference on Learning Representations*. OpenReview.net.

Ren, S., He, K., Girshick, R., and Sun, J. (2015). Faster R-CNN: Towards real-time object detection with region proposal networks. In *Neural Information Processing Systems*.

Ren, X. and Philipose, M. (2009). Egocentric recognition of handled objects: Benchmark and analysis. In *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, pages 1–8.

Rennie, C., Shome, R., Bekris, K. E., and Souza, A. F. D. (2016). A dataset for improved rgbd-based object detection and pose estimation for warehouse pick-and-place. *IEEE Robotics and Automation Letters*, 1(2):1179–1185.

Rifai, S., Vincent, P., Muller, X., Glorot, X., and Bengio, Y. (2011). Contractive auto-encoders: Explicit invariance during feature extraction. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, ICML'11, pages 833–840, USA. Omnipress.

Ronneberger, O., P.Fischer, and Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention*, volume 9351 of *LNCS*, pages 234–241. Springer. (available on arXiv:1505.04597 [cs.CV]).

Rozsa, A., Günther, M., and Boult, T. E. (2017). Are accuracy and robustness correlated? *IEEE International Conference on Machine Learning and Applications Workshops*, abs/1610.04563.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C., and Fei-Fei, L. (2015). ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision*, 115(3):211–252.

Rusu, A. A., Rabinowitz, N. C., Desjardins, G., Soyer, H., Kirkpatrick, J., Kavukcuoglu, K., Pascanu, R., and Hadsell, R. (2016). Progressive neural networks. *CoRR*, abs/1606.04671.

Sabour, S., Frosst, N., and Hinton, G. (2017). Dynamic routing between capsules. In *Advances in Neural Information Processing Systems 30*.

Saenko, K., Kulis, B., Fritz, M., and Darrell, T. (2010). Adapting visual category models to new domains. In *Proceedings of the 11th European Conference on Computer Vision: Part IV*, ECCV'10, pages 213–226, Berlin, Heidelberg. Springer-Verlag.

Sak, H., Senior, A. W., and Beaufays, F. (2014). Long short-term memory recurrent neural network architectures for large scale acoustic modeling. In *INTERSPEECH*, pages 338–342.

Savarese, S. and Li, F. F. (2007). 3d generic object categorization, localization and pose estimation. In *IEEE 11th International Conference on Computer Vision*, pages 1–8.

Schlichtkrull, M. S., Kipf, T. N., Bloem, P., van den Berg, R., Titov, I., and Welling, M. (2018). Modeling relational data with graph convolutional networks. In *The Semantic Web - 15th International Conference, ESWC 2018, Heraklion, Crete, Greece, June 3-7, 2018, Proceedings*, pages 593–607.

Schroff, F., Kalenichenko, D., and Philbin, J. (2015). Facenet: A unified embedding for face recognition and clustering. In *IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, June 7-12*, pages 815–823.

Schwartz, E., Karlinsky, L., Shtok, J., Harary, S., Marder, M., Kumar, A., Feris, R., Giryes, R., and Bronstein, A. (2018). Delta-encoder: an effective sample synthesis method for few-shot object recognition. In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R., editors, *Advances in Neural Information Processing Systems 31*, pages 2845–2855. Curran Associates, Inc.

Seff, A., Beatson, A., Suo, D., and Liu, H. (2017). Continual learning in generative adversarial nets. *CoRR*, abs/1705.08395.

Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R., and Lecun, Y. (2015). Overfeat: Integrated recognition, localization and detection using convolutional networks. *International Conference on Learning Representations*.

Shelhamer, E., Long, J., and Darrell, T. (2015). Fully convolutional networks for semantic segmentation. *Conference on Computer Vision and Pattern Recognition*, abs/1605.06211.

Shi, X., Chen, Z., Wang, H., Yeung, D., Wong, W., and Woo, W. (2015). Convolutional LSTM network: A machine learning approach for precipitation nowcasting. *Advances in Neural Information Processing Systems 28*, abs/1506.04214.

Shin, H., Lee, J. K., Kim, J., and Kim, J. (2017). Continual learning with deep generative replay. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan,

S., and Garnett, R., editors, *Advances in Neural Information Processing Systems 30*, pages 2990–2999. Curran Associates, Inc.

Simonyan, K. and Zisserman, A. (2014a). Two-stream convolutional networks for action recognition in videos. *CoRR*, abs/1406.2199.

Simonyan, K. and Zisserman, A. (2014b). Two-stream convolutional networks for action recognition in videos. In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N. D., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems 27*, pages 568–576. Curran Associates, Inc.

Simonyan, K. and Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. *International Conference on Learning Representations*, abs/1409.1556.

Singh, A., Sha, J., S. Narayan, K., Achim, T., and Abbeel, P. (2014). Bigbird: A large-scale 3d database of object instances. In *International Conference on Robotics and Automation*, pages 509–516.

Snell, J., Swersky, K., and Zemel, R. S. (2017). Prototypical networks for few-shot learning. *Conference on Neural Information Processing Systems 30*, abs/1703.05175.

Sohn, K. (2016). Improved deep metric learning with multi-class n-pair loss objective. In Lee, D. D., Sugiyama, M., Luxburg, U. V., Guyon, I., and Garnett, R., editors, *Advances in Neural Information Processing Systems 29*, pages 1857–1865. Curran Associates, Inc.

Song, H. O., Jegelka, S., Rathod, V., and Murphy, K. (2017). Deep metric learning via facility location. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2206–2214.

Song, H. O., Xiang, Y., Jegelka, S., and Savarese, S. (2016). Deep metric learning via lifted structured feature embedding. In *Computer Vision and Pattern Recognition*.

Springenberg, J. T., Dosovitskiy, A., Brox, T., and Riedmiller, M. A. (2015). Striving for simplicity: The all convolutional net. *International Conference on Learning Representations Workshops*, abs/1412.6806.

Su, J., Vasconcellos Vargas, D., and Kouichi, S. (2017). One pixel attack for fooling deep neural networks. *IEEE Transactions on Evolutionary Computation, Vol.23 , Issue.5*, pages 828–841.

Sung, F., Yang, Y., Zhang, L., Xiang, T., Torr, P. H., and Hospedales, T. M. (2018). Learning to compare: Relation network for few-shot learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Szegedy, C., Ioffe, S., and Vanhoucke, V. (2016a). Inception-v4, inception-resnet and the impact of residual connections on learning. *Thirty-First AAAI Conference on Artificial Intelligence*.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. (2015a). Going deeper with convolutions. In *Computer Vision and Pattern Recognition*.

Szegedy, C., Reed, S., Erhan, D., and Anguelov, D. (2015b). Scalable, high-quality object detection. Technical report, arXiv.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., and Wojna, Z. (2016b). Rethinking the inception architecture for computer vision. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2818–2826.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I. J., and Fergus, R. (2013). Intriguing properties of neural networks. *CoRR*, abs/1312.6199.

Taigman, Y., Yang, M., Ranzato, M., and Wolf, L. (2014). Deepface: Closing the gap to human-level performance in face verification. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1701–1708.

Tan, M. and Le, Q. (2019). EfficientNet: Rethinking model scaling for convolutional neural networks. In Chaudhuri, K. and Salakhutdinov, R., editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 6105–6114, Long Beach, California, USA. PMLR.

Tejani, A., Tang, D., Kouskouridas, R., and Kim, T.-K. (2014). Latent-class hough forests for 3d object detection and pose estimation. In Fleet, D., Pajdla, T., Schiele, B., and Tuytelaars, T., editors, *European Conference on Computer Vision*, pages 462–477, Cham. Springer International Publishing.

Tran, D., Bourdev, L. D., Fergus, R., Torresani, L., and Paluri, M. (2014). C3D: generic features for video analysis. *CoRR*, abs/1412.0767.

Tran, L., Yin, X., and Liu, X. (2017). Disentangled representation learning gan for pose-invariant face recognition. In *In Proceeding of IEEE Computer Vision and Pattern Recognition*, Honolulu, HI.

Triki Rannen, A., Aljundi, R., Blaschko, M. B., and Tuytelaars, T. (2017). Encoder based lifelong learning. *IEEE International Conference of Computer Vision*.

Tripathi, S., Lipton, Z. C., Belongie, S. J., and Nguyen, T. Q. (2016). Context matters: Refining object detection in video with recurrent neural networks. *The British Machine Vision Conference*, abs/1607.04648.

Urban, G., Geras, K. J., Kahou, S. E., Aslan, Ö., Wang, S., Mohamed, A., Philipose, M., Richardson, M., and Caruana, R. (2017). Do deep convolutional nets really need to be deep and convolutional? *5th International Conference on Learning Representations, Toulon, France, April 24-26, 2017, Conference Track Proceedings*.

Vassileios Balntas, Edgar Riba, D. P. and Mikolajczyk, K. (2016). Learning local feature descriptors with triplets and shallow convolutional neural networks. In Richard C. Wilson, E. R. H. and Smith, W. A. P., editors, *Proceedings of the British Machine Vision Conference*, pages 119.1–119.11. BMVA Press.

Venkatesan, R., Venkateswara, H., Panchanathan, S., and Li, B. (2017). A strategy for an uncompromising incremental learner. *CoRR*, abs/1705.00744.

Vijay, B., Alex, K., and Roberto, C. (2017). Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Vinyals, O., Blundell, C., Lillicrap, T., Kavukcuoglu, K., and Wierstra, D. (2016). Matching networks for one shot learning. In *Advances in Neural Information Processing Systems 29*.

Wagner, R., Thom, M., Schweiger, R., Palm, G., and Rothermel, A. (2013). Learning convolutional neural networks from few samples. *International Joint Conference on Neural Networks*.

Wang, F., Xiang, X., Cheng, J., and Yuille, A. L. (2017a). Normface: L2 hypersphere embedding for face verification. In *Proceedings of the 25th ACM International Conference on Multimedia*, MM '17, pages 1041–1049, New York, NY, USA. ACM.

Wang, H., Wang, Y., Zhou, Z., Ji, X., Gong, D., Zhou, J., Li, Z., and Liu, W. (2018). Cosface: Large margin cosine loss for deep face recognition. In *IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, June 18-22, 2018*, pages 5265–5274.

Wang, J., Zhou, F., Wen, S., Liu, X., and Lin, Y. (2017b). Deep metric learning with angular loss. In *IEEE International Conference on Computer Vision, Venice, Italy, October 22-29*, pages 2612–2620.

Wang, X., Cai, Z., Gao, D., and Vasconcelos, N. (2019). Towards universal object detection by domain attention. In *The IEEE Conference on Computer Vision and Pattern Recognition*.

Wang, X., Ding, X., and Liu, C. (2005). Gabor filters-based feature extraction for character recognition. *Pattern Recognition.*, 38(3):369–379.

Wang, X., Eliott, F. M., Ainooson, J., Palmer, J. H., and Kunda, M. (2017c). An object is worth six thousand pictures: The egocentric, manual, multi-image (emmi) dataset. In *The IEEE International Conference on Computer Vision Workshops*.

Wang, X., Gao, L., Wang, P., Sun, X., and Liu, X. (2018). Two-stream 3-d convnet fusion for action recognition in videos with arbitrary size and length. *IEEE Transactions on Multimedia*, 20(3):634–644.

Wang, Y.-X. and Hebert, M. (2016). Learning from small sample sets by combining unsupervised meta-training with cnns. In Lee, D. D., Sugiyama, M., Luxburg, U. V., Guyon, I., and Garnett, R., editors, *Advances in Neural Information Processing Systems 29*, pages 244–252. Curran Associates, Inc.

Wang, Z., He, K., Fu, Y., Feng, R., Jiang, Y.-G., and Xue, X. (2017d). Multi-task deep neural network for joint face recognition and facial attribute prediction. In *Proceedings of the 2017 ACM on International Conference on Multimedia Retrieval*, pages 365–374, New York, NY, USA. ACM.

Wen, Y., Zhang, K., Li, Z., and Qiao, Y. (2016). A discriminative feature learning approach for deep face recognition. In Leibe, B., Matas, J., Sebe, N., and Welling, M., editors, *European Conference on Computer Vision*, volume 9911 of *Lecture Notes in Computer Science*, pages 499–515. Springer.

Wistuba, M., Rawat, A., and Pedapati, T. (2019). A survey on neural architecture search. *CoRR*, abs/1905.01392.

Worrall, D. E., Garbin, S. J., Turmukhambetov, D., and Brostow, G. J. (2017). Harmonic networks: Deep translation and rotation equivariance. *Conference on Computer Vision and Pattern Recognition*, abs/1612.04642.

Wu, C.-Y., Manmatha, R., Smola, A. J., and Krähenbühl, P. (2017). Sampling matters in deep embedding learning. *2017 IEEE International Conference on Computer Vision*, pages 2859–2867.

Wulfmeier, M., Bewley, A., and Posner, I. (2017). Addressing appearance change in outdoor robotics with adversarial domain adaptation. In *International Conference on Intelligent Robots and Systems*, pages 1551–1558. IEEE.

Xian, Y., Lorenz, T., Schiele, B., and Akata, Z. (2018). Feature generating networks for zero-shot learning. In *IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, June 18-22*, pages 5542–5551.

Xiao, T., Zhang, J., Yang, K., Peng, Y., and Zhang, Z. (2014). Error-driven incremental learning in deep convolutional neural network for large-scale image classification. In *Association for Computing Machinery Multimedia*.

Xie, S., Girshick, R., Dollár, P., Tu, Z., and He, K. (2017). Aggregated residual transformations for deep neural networks. *Conference on Computer Vision and Pattern Recognition*.

Xie, S., Kirillov, A., Girshick, R. B., and He, K. (2019). Exploring randomly wired neural networks for image recognition. *CoRR*, abs/1904.01569.

Xing, E. P., Ng, A. Y., Jordan, M. I., and Russell, S. (2002). Distance metric learning, with application to clustering with side-information. In *Proceedings of the 15th International Conference on Neural Information Processing Systems*, pages 521–528, Cambridge, MA, USA. MIT Press.

Yi, D., Lei, Z., Liao, S., and Li, S. Z. (2014). Learning face representation from scratch. *CoRR*, abs/1411.7923.

Yin, X. and Liu, X. (2018). Multi-task convolutional neural network for pose-invariant face recognition. *IEEE Transactions on Image Processing*, 27(2):964–975.

Yin, X., Yu, X., Sohn, K., Liu, X., and Chandraker, M. (2019). Feature transfer learning for face recognition with under-represented data. In *The IEEE Conference on Computer Vision and Pattern Recognition*.

Yosinski, J., Clune, J., Bengio, Y., and Lipson, H. (2014). How transferable are features in deep neural networks? In *Proceedings of the 27th International Conference on Neural Information Processing Systems*, pages 3320–3328, Cambridge, MA, USA. MIT Press.

Zamir, A. R., Wu, T., Sun, L., Shen, W., Malik, J., and Savarese, S. (2017). Feedback networks. *Conference on Computer Vision and Pattern Recognition*, abs/1612.09508.

Zeiler, M. D. and Fergus, R. (2014). Visualizing and understanding convolutional networks. *European Conference on Computer Vision*, abs/1311.2901.

Zeng, A., Song, S., Yu, K.-T., Donlon, E., Hogan, F. R., Bauza, M., Ma, D., Taylor, O., Liu, M., Romo, E., Fazeli, N., Alet, F., Dafle, N. C., Holladay, R., Morona, I., Nair, P. Q., Green, D., Taylor, I., Liu, W., Funkhouser, T., and Rodriguez, A. (2018). Robotic pick-and-place of novel objects in clutter with multi-affordance grasping and cross-domain image matching. In *Proceedings of the IEEE International Conference on Robotics and Automation*.

Zeng, A., Yu, K.-T., Song, S., Suo, D., Walker Jr, E., Rodriguez, A., and Xiao, J. (2017). Multi-view self-supervised deep learning for 6d pose estimation in the amazon picking challenge. In *Proceedings of the IEEE International Conference on Robotics and Automation*.

Zenke, F., Poole, B., and Ganguli, S. (2017). Continual learning through synaptic intelligence. In Precup, D. and Teh, Y. W., editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 3987–3995, International Convention Centre, Sydney, Australia. PMLR.

Zhang, C., Bengio, S., Hardt, M., Recht, B., and Vinyals, O. (2017a). Understanding deep learning requires rethinking generalization. *International Conference on Learning Representations*, abs/1611.03530.

Zhang, J., Zheng, Y., and Qi, D. (2017b). Deep spatio-temporal residual networks for citywide crowd flows prediction. *Thirty-First AAAI Conference on Artificial Intelligence*, abs/1610.00081.

Zhang, W., Ouyang, W., Li, W., and Xu, D. (2018). Collaborative and adversarial network for unsupervised domain adaptation. In *The IEEE Conference on Computer Vision and Pattern Recognition*.

Zhe, X., Chen, S., and Yan, H. (2018). Directional statistics-based deep metric learning for image classification and retrieval. *Pattern Recognition*, 93.

Zheng, S., Song, Y., Leung, T., and Goodfellow, I. (2016). Improving the robustness of deep neural networks via stability training. In *Conference on Computer Vision and Pattern Recognition*.

Zhi, S., Liu, Y., Li, X., and Guo, Y. (2017). Toward real-time 3d object recognition: A lightweight volumetric cnn framework using multitask learning. *Computers & Graphics*, 71.

Zhou, B., Khosla, A., Lapedriza, À., Oliva, A., and Torralba, A. (2016). Learning deep features for discriminative localization. *Conference on Computer Vision and Pattern Recognition*, abs/1512.04150.

Zhou, K. (2018). Pytorch implementation of center loss. `https://github.com/KaiyangZhou/pytorch-center-loss`.

Zhu, Y., Elhoseiny, M., Liu, B., Peng, X., and Elgammal, A. (2018). A generative adversarial approach for zero-shot learning from noisy texts. In *The IEEE Conference on Computer Vision and Pattern Recognition*.

Zoph, B., Vasudevan, V., Shlens, J., and Le, Q. V. (2018). Learning transferable architectures for scalable image recognition. *Conference on Computer Vision and Pattern Recognition*, abs/1707.07012.